Book in progress
Tecnologie Informatiche Open Office

Il Progetto Book in Progress

La rete Book in Progress è nata per produrre libri di testo, di elevato spessore scientifico e comunicativo, scritti dagli stessi docenti, appartenenti a più di cento scuole del secondo ciclo e del primo ciclo della rete nazionale, con istituto capofila l'ITIS "E. Majorana" di Brindisi.

Tale iniziativa migliora significativamente l'apprendimento degli allievi e contemporaneamente fornisce una risposta concreta ai problemi economici delle famiglie e del caro libri.

Con tale iniziativa, si valorizza la funzione docente e si concretizza la personalizzazione degli interventi formativi.

I testi prodotti sono disponibili in formato cartaceo ed in alcuni formati elettronici come pdf, epub e ibooks . Altro elemento positivo, che tale iniziativa porta con se, è quello di consenti-re alle famiglie un risparmio di spesa sulla dotazione libraria di circa €300,00 rispetto ai tetti di spesa previsti dal Ministero.

Nel libro **"Tecnologie Informatiche" – EDIZIONE 2018** è stato completamente rinnovato **capitolo 6 "Algoritmi e Diagrammi di Flusso"**, inoltre sono state recepite le richieste per-venute da alcune scuole della rete ed è stata realizzato il nuovo **capitolo 11 "Creare App con APPINVENTOR"** dai nuovi collaboratori Grazia Marzia e Angelo Caputo. Altra novità è l'inserimento di QR code per accedere direttamente con dispositivi **BYOD** come smartphone o tablet ai contenuti aggiuntivi multimediali e ad altre risorse sulla rete.

Si ringraziano gli autori:
Prof. **Angelo Oliva** – I.I.S.S "Ettore Majorana" – Martina Franca (TA)
Prof. **Dario Rinaudo**, Prof. **Domenico Deluso**, Prof.ssa **Graziella Locatelli** – I.S.I.S. "Gu-glielmo Oberdan" – TREVIGLIO (BG)
Prof. **Enrico Sartirana** – I.T.I.S. "Stanislao Cannizzaro" Rho (MI)
Prof. **Salvatore Madaro**, Prof.ssa **Giorgia Martina** – I.T.I.S "E. Fermi" – Francavilla Fonta-na (BR)
Prof. **Paolo Lillo** – I.S.I.S. "L. Scarambone" – Lecce
Prof.ssa **Alessandra D'Orazio** – I.S.I.S. "Alessandro Volta" – Frosinone
Prof. **Grazia Marzia e Angelo Caputo** – I.I.S.S "C. Mondelli" – Massafra (TA)
Prof. **Luca Peresson** dell'I.T.I.S "A. Malignani" – Udine

Rielaborazione unità didattiche, riduzione e impaginazione a cura del Prof. Angelo Oliva dell' I.I.S.S. "Ettore Majorana " – Martina Franca (TA) con la preziosa collaborazione del Prof. Dario Rinaudo dell' I.S.I.S. "Guglielmo Oberdan" – TREVIGLIO (BG)

Maggio 2019

Rete Nazionale Book in Progress
Dipartimento Disciplinare d'Informatica
Coordinatore Nazionale
Prof. Angelo Oliva

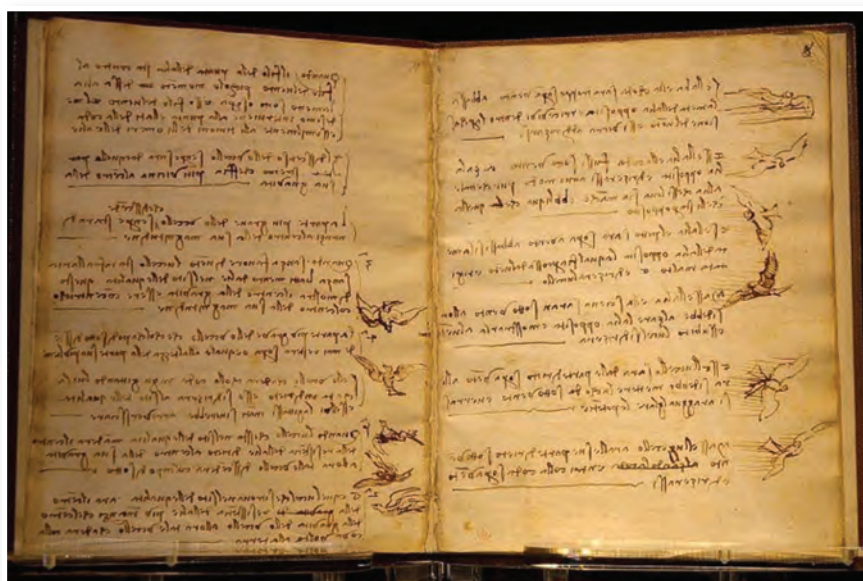
INDICE

1. Concetti di base dell'ICT

Autore: **Angelo Oliva**
I contenuti delle nozioni fondamentali dellinformatica sono rielaborati dall'unità didattica a cura di **DARIO RINAUDO**

Competenze	Abilità	Conoscenze
Utilizzare linguaggio tecnico in merito a componenti e problematiche di funzionamento del personal computer e in generale sul mondo delle tecnologie informatiche	*Riconoscere le componenti interne ed esterne del personal computer*	*Nozioni storiche, generali e fondamentali della Tecnologia dell'informazione e della comunicazione*

Introduzione all'informazione e alla comunicazione

Quante volte sarà stato scoperto il fuoco prima che questo sia stato sistematicamente utilizzato dall'uomo? Chi sono stati i primi uomini a usare la scrittura? l'inchiostro? la polvere da sparo? Probabilmente in più luoghi della terra sono state fatte le stesse scoperte e le stesse invenzioni, senza che gli autori fossero a conoscenza delle esperienze già fatte da altri. Le scoperte scientifiche e le invenzioni erano destinate a scomparire con i loro inventori se queste non fossero comunicate in qualche modo all'umanità, ecco perché la comunicazione è stata un processo fondamentale per il progresso. *La comunicazione è un processo di trasmissione d'informazioni che prevede l'esistenza di una sorgente emittente e di un ricevente o destinatario e infine l'utilizzo di un canale, cioè di un mezzo di trasmissione.* Il tipo di canale utilizzato è fondamentale sull'esito della comunicazione: se si usa la voce e l'etere si ha sicuramente un numero limitato di destinatari, mentre un messaggio stampato o trasmesso via radio avrà potenzialmente un numero maggiore di destinatari.
Non sempre il solo mezzo garantisce la diffusione dell'informazione, basti pensare ai progetti delle macchine di Leonardo da Vinci, di cui l'inventore ha lasciato accurata documentazione su carta, ma ciò nonostante, tali progetti sono rimasti sconosciuti per secoli a causa anche della modalità che Leonardo utilizzava nella scrittura, detta speculare perché consiste nello scrivere le lettere come se fossero riflesse da uno specchio. Con l'invenzione della stampa, si diffusero libri e pubblicazioni scientifiche, che consentirono agli scienziati di condividere le informazioni sui propri esperimenti, iniziò così quel processo che portò sempre più rapidamente alle attuali tecnologie.

L. Da Vinci codice sul volo degli uccelli – Fotografo Luc Viatour - www.lucnix.be - CC BY-SA 3.0

T.I.C. e competenze digitali

Sebbene sia nato per eseguire calcoli complessi, il computer oggi è diventato uno strumento rapido e flessibile per rappresentare, gestire e comunicare informazioni. Le molteplici attività legate all'uso delle più recenti apparecchiature informatiche sono identificate col termine **TIC**, acronimo ottenuto da *Tecnologie dell'Informazione e della Comunicazione, che poi deriva dalla traduzione di Information and Communication Technology* e dal corrispondente acronimo **ICT.**
La diffusione della rete internet e di dispositivi come smartphone, tablet e Personal Computer in tutte le case, consente, e a volte rende obbligatorio, il loro utilizzo non solo per svago ma anche per gestire molteplici attività come la gestione del conto corrente bancario o il rapporto con la Pubblica Amministrazione, al punto che si parla oggi di **Home Office** ("ufficio casalingo"). Per questo si rende necessario per il cittadino di oggi possedere tra le altre competenze chiave anche le competenze digitali.

La raccomandazione del Parlamento e del Consiglio europei del 18 dicembre 2006 dell' Unione Europea definisce le otto competenze chiave per l'apprendimento permanente "Key Competences for Lifelong Learning" e tra queste la quarta è la Competenza digitale che:
«Consiste nel saper utilizzare con dimestichezza e spirito critico le tecnologie della società dell' informazione per il lavoro, il tempo libero e la comunicazione.
Essa è supportata da abilità di base nelle Tic: l'uso del computer per reperire, valutare, conservare, produrre, presentare e scambiare informazioni, nonché per comunicare e partecipare a reti collaborative tramite Internet».

Il cittadino che si avvale di questi servizi elettronici, oggi detto **e-citizen**, troverà sempre più spesso dei vantaggi anche sotto l'aspetto economico.

"Key Competences for Lifelong Learning"

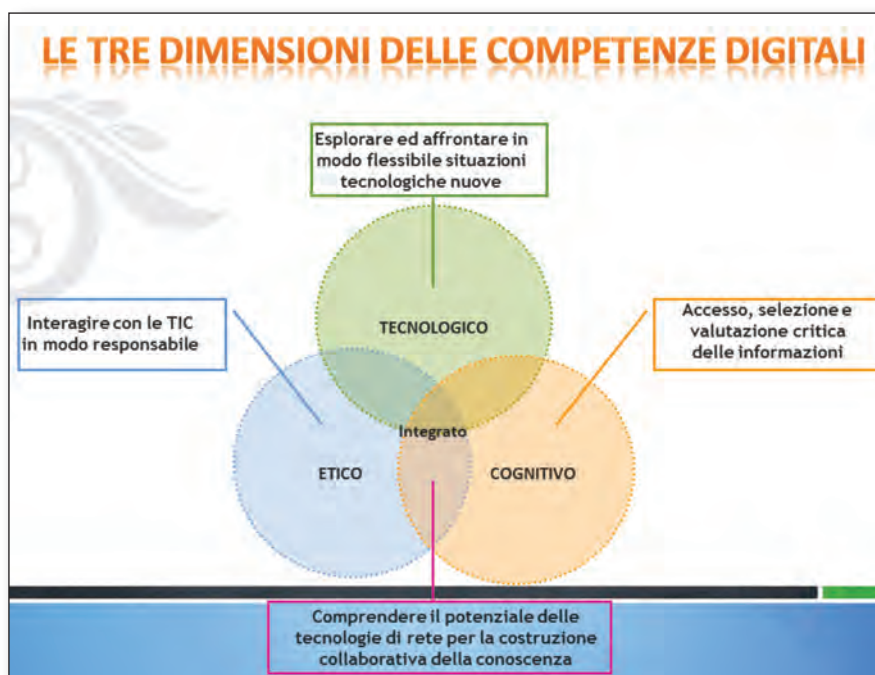
1) Communication in the mother tongue;
2) Communication in foreign languages;
3) Mathematical competence and basic competences in science and technology;
4) Digital competence;
5) Learning to learn;
6) Social and civic competences;
7) Sense of initiative and entrepreneurship;
8) Cultural awareness and expression.

http://ec.europa.eu/education/policies/2010/doc/keyrec_it.pdf

Gli studi in corso hanno portato alla strutturazione della competenza digitale in tre dimensioni:

• **dimensione tecnologica:**
riconoscere le criticità tecnologiche e le interfacce, selezionare la tecnologia adeguata per ciascun compito, operare logicamente, rappresentare processi simbolici, distinguere tra reale e virtuale.

• **dimensione cognitiva:**
saper trattare ovvero sintetizzare, rappresentare e analizzare testi, dati, tabelle e grafici e saper valutare la pertinenza dell'informazione e la sua affidabilità.

• **dimensione etica:**
conoscere i concetti di tutela della privacy, rispettare i diritti intellettuali dei materiali reperiti in Internet e l'immagine degli altri, comprendere il dislivello sociale e tecnologico che può esistere tra paesi, persone, generazioni, e il problema dell'accessibilità

Un po' di storia: dai grandi elaboratori al personal computer.

Il termine "Computer deriva dal latino "computo" (contare). In effetti, si tratta di una macchina specializzata per eseguire calcoli. Ispirandosi all'**abaco Blaise Pascal** nel 1642, inventò la pascalina, uno strumento meccanico in grado di eseguire addizioni con riporto.

La Pascaline, esposta al Musée des Arts et Métiers, di Parigi, Fotografia di David Monniaux

Charles Babbage realizzò nel 1850 una macchina capace di ricevere una serie di istruzioni che avrebbe potuto svolgere anche non immediatamente ma successivamente. Fu questo lo spunto per realizzare macchine adibite all'esecuzione di operazioni ripetitive con l'utilizzo di schede, che trovarono grande applicazione nel campo dell'industria tessile.

Durante la seconda guerra mondiale lo sviluppo si intensificò, grazie alle intuizioni di scienziati come **Turing** e **Von Neumann**, arrivando alla realizzazione, nell'immediato dopoguerra, dei primi elaboratori il **Mark1** e **l'ENIAC.**

La costruzione del **Mark1** iniziò nel 1939 al dipartimento di fisica dell'Università di Harvard (Cambridge, Massachusetts, U.S.A.) e terminò nel 1943 presso i laboratori dell'IBM ad ENDICOTT (New York), che finanziò il progetto per un importo finale di 250.000 dollari. Esso era costituito da 765.000 componenti, interruttori, relè, alberi di rotazione e frizioni e centinaia di chilometri di cavi, occupava una lunghezza di 16 m, si sviluppava in altezza per 2,4 m ed aveva una profondità di circa 0,5 metri; pesava circa 4 tonnellate e mezzo. Il Mark I poteva memorizzare 72 numeri di 23 cifre decimali ciascuno. Poteva eseguire tre addizioni o sottrazioni al secondo, una moltiplicazione in 6 secondi, una divisione in 15,3 secondi ed un logaritmo oppure una funzione trigonometrica in più di un minuto. Il Mark I leggeva le sue istruzioni su delle schede perforate e, eseguita l'istruzione corrente, passava alla successiva. I programmi complessi erano fisicamente lunghi.

Mark I - unità Input-Output CC BY-SA 3.0
Caricato da Daderot su en.wikipedia

L'ENIAC al Ballistic Research Laboratory Philadelphia, Pennsylvania

L'ENIAC fu progettato e costruito alla Moore School of Electrical Engineering un'ex scuola universitaria dell'Università della Pennsylvania, allo scopo di realizzare di una macchina da calcolo capace di risolvere i problemi di calcolo balistico per il lancio dei proiettili di artiglieria. Per la sua realizzazione, terminata nel 1946 con una spesa complessiva otto volte maggiore di quella preventivata e pari a circa 486.800 dollari, furono necessarie ben 18.000 valvole termoioniche che portarono l'ambiente ad una temperatura superiore ai 50 °C. Purtroppo, infatti, il transistor che soppianterà la valvola termoionica nella realizzazione del computer elettronico fu inventato solo un anno. L'ENIAC occupava una superficie di 167 mq e pesava oltre 30 tonnellate, pur eseguendo calcoli elementari e assorbiva così tanta energia elettrica che, alla sua prima messa in funzione, causò un black-out nel quartiere ovest di Filadelfia. John Von Neumann lo utilizzò per eseguire la prima previsione metereologica al computer. L'ENIAC rimase in funzione fino al 2 ottobre 1955, fu poi trasferito a Washington, al museo Smithsonian Institution, dove è ancora esposto.

Jhon Von Neumann - Budapest 28/12/1903 Washington 8/02/1957

Lo scienziato **John Von Neumann** nel 1946 pubblicò un articolo sul progetto di una macchina per il calcolo automatico che introduceva il concetto di programma registrato in memoria. Il calcolatore digitale moderno nasce all'inizio degli anni '40 per merito di John von Neumann, J.Presper Eckert e John William Mauchly. Questo team di scienziati concettualizzò l'idea di macchina con accesso casuale e non sequenziale ai dati in memoria

(RAM Random Access Machine). Il calcolatore ha ancora una memoria organizzata come sequenza di celle d'informazione, ma il tempo di accesso a una cella non dipende dalla più dalla posizione della cella stessa, in pratica non si devono scorrere le celle sino a trovare quella cercata, ma si accede a essa direttamente.

Tale macchina, nota come **Modello di Von Neumann** è stata ed è tuttora l'architettura su cui si basano i moderni calcolatori. La sua caratteristica fondamentale che la memoria contiene sia i **programmi** sia i **dati**.

Il programma è:
- ✓ la sequenza d'istruzioni che devono essere eseguite
- ✓ i dati sono le informazioni che il programma elabora per produrre il risultato.

Queste informazioni includono:
- ✓ i dati d'ingresso
- ✓ i dati di lavoro, cioè quei dati parziali ottenuti durante l'elaborazione
- ✓ i risultati della computazione.

Questa copresenza di dati e programmi nella stessa memoria di lavoro rappresenta la fondamentale differenza tra le macchine costruite o pensate sino a quel momento e i moderni calcolatori digitali. Da questo momento, infatti, è possibile che un programma possa scrivere nell'area di memoria di un altro un altro programma, consentendo una nuova serie di tecniche di programmazione importantissime, come quella detta del "Produttore Consumatore" dove appunto una procedura o programma può generare dati che vengono utilizzati da un altro programma anche ciclicamente.

La macchina Von Neumann

Gli elementi funzionali della Macchina sono quattro:

- ✓ **Unità di elaborazione (CPU, Central Processor** Unit) dove vengono eseguite le istruzioni una per volta dei processi in esecuzione
- ✓ **Memoria centrale** dove vengono caricati dati iniziali, temporanei e programmi durante il funzionamento,
- ✓ **Periferiche** le unità che consentono di immettere dati nel sistema e di ricevere messaggi e risultati in uscita
- ✓ **Bus di sistema** il canale che collega tutti i componenti del sistema, permettendo loro di dialogare

Nel 1962, Philippe Dreyfus conia il termine **informatica**, dalle parole ***inform**ation electronique ou autom**atique***, per definire il trattamento automatico dell'informazione mediante calcolatore.

L'uso dei transistor e la miniaturizzazione consentirono la realizzazione, già negli anni '70, dei primi **Home Computer**, computer casalinghi per applicazioni molto elementari, la gestione di piccoli archivi e principalmente per videogiochi, molto noti all'epoca erano i Commodore VIC 20 e il Commodore CBM 64, dove 64 indicava i Kilobyte di Ram, il Sinclair Spectrum, e l'Amiga di Atari. Nel 1981 arrivano sul mercato i primi **Personal Computer IBM**, che iniziarono a soddisfare le esigenze di piccole aziende nelle classiche operazioni di corrispondenza, calcolo, disegno tecnico, fino alla gestione dell'intera azienda.

Un esemplare della Programma 101 esposta al Museo della toria del Computer di Mountain View – Fotografo Bill Abbott: Olivetti printing calculator/computer su Flickr CC BY-SA 2.0

Non si può dimenticare **il contributo italiano all'informatica**, la Società Olivetti sotto la guida di Adriano Olivetti nel 1940 produsse la sua prima macchina addizionatrice Olivetti, seguita nel 1945 dalla Divisumma 14, la prima calcolatrice scrivente al mondo in grado di eseguire le quattro operazioni, queste erano macchine completamente meccaniche. Nel 1959 Olivetti sviluppa

uno dei primi mainframe computer a transistor: **Elea 9003**. Il prodotto però più rivoluzionario di Olivetti risulterà, la **Programma 101** ritenuta oggi il primo personal computer, progettato dall'ing. **Pier Giorgio Perotto** e presentato insieme alla Logos 27, un'efficientissima calcolatrice meccanica, alla fiera di New York del 1965. Probabilmente neanche l'azienda poteva prevedere il potenziale e il successo di tale prodotto tanto che pur disponendo di un grandioso stand allestito per la Logos, la P101 venne relegato in una saletta di fondo, dopo che i primi visitatori si accorsero delle caratteristiche della P101, l'afflusso di visitatori fu tale che il personale dello stand dovette improvvisare un servizio d'ordine per l'accesso allo stand. Ciò che colpiva della P101 era la capacità per una macchina che poteva stare sopra una scrivania, da cui il termine desktop computer, di eseguire operazioni, allora piuttosto complesse. La rete televisiva americana NBC acquistò 5 esemplari per computare i risultati elettorali da fornire ai propri telespettatori. Fu l'inizio di un successo mondiale, si ricorda che il deposito del brevetto di alcune soluzioni tecniche della P101 obbligò la Hewlett-Packard meglio nota come HP, a riconoscere un indennizzo di 900.000 dollari all'Olivetti quando le fu contestata la violazione di brevetto. Della P101 furono vendute circa 44.000 unità.

Anche un altro grande personaggio italiano dell'informatica ha iniziato la sua carriera nell'Olivetti, il suo contributo in questo campo è stato così importante che potrebbe essere celebrato al pari di Bill Gates e Stevie Jobs, si tratta di **Federico Faggin** nato a Vicenza nel 1941 ed è un imprenditore, inventore e informatico italiano naturalizzato statunitense. Faggin nel 1970 è stato il capo progetto dell'Intel 4004, primo microprocessore al mondo, e dei successori Intel 8008, 4040 e 8080. Fu anche lo sviluppatore della tecnologia MOS con porta di silicio che permise la fabbricazione dei primi microprocessori e delle memorie EEPROM e RAM dinamiche. Nel 1974 Faggin fondò e diresse la ditta Zilog dove dette vita al famoso microprocessore Z80 ancora usato nel 2013. Nel 1986 Faggin fondò e diresse la Synaptics, ditta che sviluppò i primi Touchpad e Touchscreen.

Federico Faggin - Vicenza, 1 dicembre 1941

Nozioni fondamentali dell'Informatica

HARDWARE

Per **hardware** s'intende tutti ciò che ha una consistenza fisica, un peso, quindi le componenti fisiche del computer compresi i dispositivi elettrici ed elettronici. La tastiera, il mouse, il monitor, le casse acustiche, l'unità centrale, ecc., fanno parte dell'hardware.

SOFTWARE

Per **software** intendiamo tutto ciò che è utile o indispensabile al computer per funzionare e non ha una consistenza fisica, quindi i programmi che servono e permettono al computer di compiere delle azioni. È software il sistema operativo, un videogioco, il programma per elaborare testi, di disegno, i driver di un dispositivo periferico come la stampante ecc.

• Il **software di base** è il sistema operativo, ne sono esempi Windows, l'analogo sistema operativo gratuito e open source Linux , Mac OS della Apple computer, Android, usato su vari dispositivi mobili come tablet e smartphone. Lo scopo del sistema operativo è di riconoscere le risorse del sistema come la quantità di memoria centrale (RAM) e dispositivi di memoria come dischi rigidi o pen-drive, e metterli a disposizione dell'utente attraverso un' interfaccia grafica utente amichevole (GUI Graphic User-friendly Interface)

• Il **software applicativo** è costituito da quella classe di programmi dedicati a svolgere perticolari applicazioni, come i software per l'editing video, il fotoritocco oltre che quei software dedicati all'office automation, gli elaboratori di testi, i fogli elettronici o di calcolo, i software per la gestione di database, tali software sono generalmente disponibili in "Suite" di cui le più note sono Microsoft Office, e gli analoghi gratuiti OpenOffice e LibreOffice.

Classificazione e distribuzione dei software

Un programma venduto e anche quelli distribuiti in varie forme, hanno la licenza d'uso (**EULA - End User Licence Agreement**, cioè l'accordo di contratto del produttore del software con l'utente finale).
Esistono in commercio oltre ai software commerciali, altri tipi di software che in genere, vengono scaricati da internet, esaminiamoli:

1) Open source (codice aperto): distribuito e copiabile gratuitamente, il cui codice sorgente è accessibile e modificabile, senza violare il diritto d'autore.

2) Freeware (libero): distribuito e duplicabile gratuitamente, il cui codice non è però accessibile.

3) Shareware (condiviso): circola liberamente sulla rete e può essere copiato ed utilizzato gratuitamente nei limiti indicati dalla licenza. Possono essere previsti l'utilizzo:
   a. temporaneo, entro un certo termine;
   b. parziale, con alcune funzioni disattivate.

4) Public domain (dominio pubblico): chi lo ha creato non esercita il diritto d'autore, per cui è gratuitamente utilizzabile e duplicabile.

Ma cos'è il DIRITTO D'AUTORE?

La diffusione del computer e di internet ha reso più pressante la questione della tutela del ***copyright*** (diritto d'autore). Infatti è noto come sia facile copiare un programma per videogame con il computer o scaricare, da internet, files di qualsiasi genere (immagini, musica,…) e software (programmi per elaboratore). La legislazione italiana sulla protezione del diritto d'autore e di altri diritti connessi al suo esercizio è stata aggiornata per comprendere anche il software e i database, equiparati ad opere dell'ingegno di carattere creativo. Gli autori di opere letterarie, composizioni musicali, disegni, opere cinematografiche e fotografiche erano tutelati nella prima emanazione della legge, oggi colui che crea un software è il titolare dei diritti esclusivi di riprodurre, distribuire, noleggiare o dare in prestito l'opera o esemplari di essa. La SIAE detiene il registro pubblico in cui è trascritto il nome del software e dell'autore che così ne può provare la paternità. Chi abusivamente utilizza, duplica, riproduce le opere munite del contrassegno della SIAE oppure acquista o noleggia attrezzature per eluderne le protezioni è punito con una multa e la confisca del materiale. Oltre la multa, rischia il carcere chi abusivamente duplica, distribuisce, vende e detiene, per trarne profitto, software o banche dati, di cui non è l'autore, oppure ne rimuove le chiavi di protezione.

Le più diffuse tipologie di computer

Oggi le più diffuse tipologie di computer sono:

Desktop PC – personal computer: Il primo PC fu lanciato in commercio dalla IBM nel 1981. E' un elaboratore monoutente cioè pensato per l'uso da parte di una persona per volta. Oggi ha raggiunto prestazioni tali da poter sostenere carichi di lavoro che in passato erano svolti da mainframe e workstation. La velocità di esecuzione già molto elevata, è aumentata ancora con l'introduzione dei processori multicore (dualcore, quadcore), dove il processore, che è l'unità che esegue l'elaborazione delle istruzioni, da l'impressione che le esecuzioni dei vari programmi avvenga contemporaneamente, ma in realtà sono eseguiti ciclicamente uno per volta. Il personal computer è efficacemente utilizzato in luoghi di lavoro o come postazione domestica sia per programmi applicativi professionali, sia ludici, sia per navigare in internet.

Laptop – computer portatile, ha raggiunto le stesse prestazioni di un Desktop PC, ma consente la mobilità perchè è fornito di batteria ed ha caratteristiche di dimensioni e peso contenuti. I computer portatili rispetto ai PC, in genere non si prestano a notevoli aggiornamenti hardware.

Netbook: è un portatile di dimensioni e prestazioni ridotte, il suo monitor ha la grandezza di pochi pollici ed è usato per semplici applicazioni e per collegarsi ad internet.

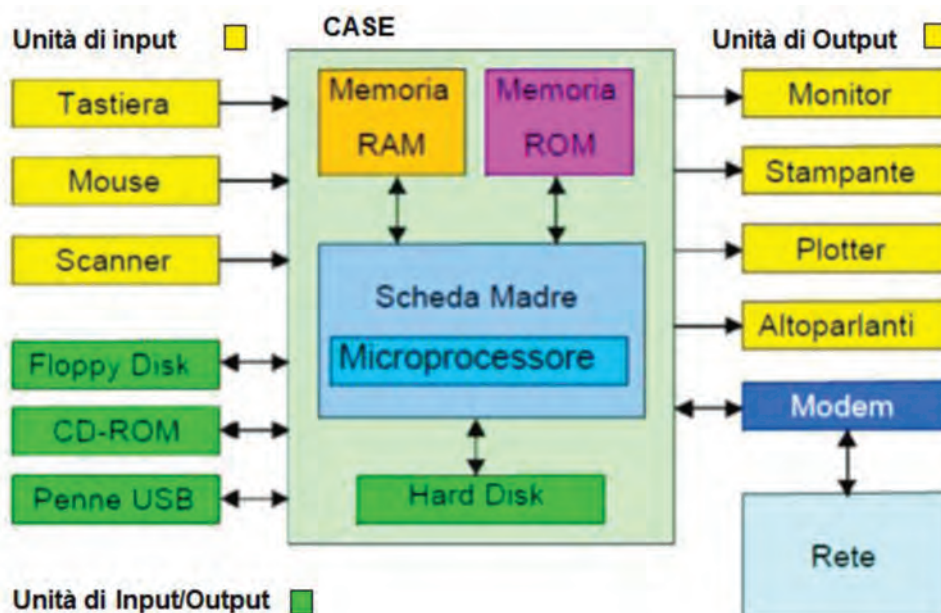
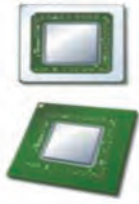
Si possono citare anche i seguenti dispositivi che non rientrano nella categoria dei computer, poiché non consentono di fare molte attività, prime fra tutte la programmazione, ma si prestano a svolgere comodamente alcune funzioni come la navigazione in internet ed in particolare nell'utilizzo dei social network e nella gestione della posta elettronica:

Tablet: con prestazioni similari a un portatile, ma di dimensione e peso contenuti, in genere non sono dotati di tastiera esterna, l'utente interagisce con tale strumento tramite touch screen. Gli schermi LCD, sono retro-illuminati per cui alla luce del sole, la lettura non è sempre agevole.
Fra gli altri dispositivi portatili bisogna ricordare lo **SMARTPHONE** che è un telefono cellulare e un semplice computer, infatti su esso possono essere installate applicazioni, si possono gestire dati, video, foto e ci si può collegare in rete.

Componenti Hardware del personal computer

L'Hardware è l'insieme delle componenti fisiche del computer, nella loro descrizione è essenziale una prima classificazione. Iniziamo dal **Case** che è quella scatola generalmente in versione Desktop Tower. Al suo interno sono presenti una serie di dispositivi per il funzionamento del computer. A esso poi si collegano una serie di dispositivi esterni che servono per immettere dati e comandi (dispositivi di input) e per ricevere i risultati delle azioni richieste (dispositivi di output), che sono informazioni di vario formato, testo, immagine, suono, video, e altro, si pensi per esempio all'effetto vibrazione di alcune periferiche di gioco.

I principali dispositivi contenuti all'interno del Case sono:
- La Scheda Madre (Mainboard)
- La CPU (Microprocessore)
- Le Porte di comunicazione
- Le Memorie
- Le Memorie di Massa

I Principali dispositivi esterni detti Unità Periferiche sono:
- Dispositivi di Input
- Dispositivi di Output
- Dispositivi di Input e Output

La **scheda madre**, spesso chiamata con il suo nome inglese motherboard, è progettata per far dialogare tra di loro le varie
componenti del pc, contiene connettori, slot, punti di alimentazione, e un insieme di circuiti integrati, detto chipset. Microprocessori e banchi di memoria Ram e varie schede tra cui quella grafica e quella audio vengono direttamente istallate su di essa, così come sul telaio di un'automobile vengono assemblati il motore e le parti meccaniche, elettriche e la carrozzeria.

La componente più importante è La **CPU** (Central Processing Unit) che può essere considerata il cervello del computer, chiamata anche processore o microprocessore. Essa svolge le sue funzioni temporizzando le sue operazioni con **l'impulso di clock**, un segnale sincrono generato da un circuito interno chiamato appunto clock. La velocità di esecuzione delle operazioni della CPU corrisponde alla velocità di emissione di questo impulso, si chiama frequenza di clock e viene misurata in hertz. Se un clock lavora a un GHz (Gigaherz), vuol dire che l'unità centrale esegue in un secondo un miliardo d'istruzioni elementari. Se un clock lavora a un MHz (Megaherz), vuol dire che l'unità centrale esegue in un secondo un milione di istruzioni elementari.

Per eseguire un qualsiasi programma La CPU estrae una per volta le istruzioni ed i dati necessari dalla RAM, con la quale comunica attraverso delle linee di collegamento chiamate **BUS**. Nella CPU è integrata la memoria cache che è mirata a rendere immediatamente disponibili le prossime istruzioni da eseguire, è una memoria velocissima, più memoria cache c'è in un PC, più alte sono le sue prestazioni. Questo tipo di memoria non è espandibile.

La CPU è sostanzialmente costituita dalle seguenti componenti:
• La **ALU** (unità logico aritmetica) che esegue le operazioni aritmetiche, le operazioni logiche.
• La **CU** (control unit) che sovraintende al coordinamento delle varie componenti hardware del microprocessore durante l'esecuzione dei processi.
•I **registri**, sui quali la CPU memorizza in modo temporaneo i dati da elaborare.

Le memorie, sono anch'esse di varie tipologie e svolgono diverse funzioni:
<u>RAM</u> (Random Access Memory): La memoria RAM è una memoria volatile, cioè perde il suo contenuto nel momento in cui non viene più alimentata dalla tensione di lavoro. Tutto il lavoro che viene introdotto nell'elaboratore, va a depositarsi sulla memoria ram fino a quando non si effettua il salvataggio sull'hard disk o su un'altra memoria permanente. Se dovesse spegnersi il computer senza avere salvato il lavoro, i dati sarebbero irrimediabilmente persi. Insieme alla CPU è uno dei dispositivi che contribuisce ad aumentare le prestazioni del computer, la ram è in genere espandibile entro dimensioni dipendenti dall'architettura della stessa macchina.

<u>ROM</u> (Read Only Memory): la memoria ROM è una memoria di sola lettura nella quale il costruttore del PC memorizza un programma di inizializzazione e controllo, il **BIOS**, che all'accensione va in esecuzione nella cosiddetta fase di bootstrap, durante la quale vengono effettuati una serie di test e riconosciute le componenti istallate come la memoria ram, le unità di memoria di massa come i dischi rigidi, le periferiche come la tastiera. Se in questa fase si verifica qualche anomalia, questa viene segnalata con una serie di beep che codificano il tipo di errore riscontrato, altrimenti si procede con il caricamento del sistema operativo.

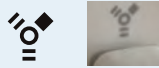
La ROM non è una memoria volatile, allo spegnimento del PC i dati memorizzati sulla ROM non vengono persi poiché sono registrati su questa memoria in modo permanente. Il costruttore può anche rilasciare versioni aggiornate e migliorate del bios che possono essere aggiornate con una particolare e delicata procedura nella versione delle ROM riscrivibili, le **EEPROM** acronimo di *Electrically Erasable*, ROM.

<u>L'HARD DISK</u> o disco rigido è un supporto magnetico dove vengono installati i programmi e dove possono essere conservati e salvati i nostri lavori. Può essere costituito da un insieme di dischi, gira a una velocità che varia secondo la capacità di memorizzazione. Sul disco si legge e si scrive mediante una testina di lettura-scrittura. Esistono in commercio dischi esterni che si collegano al computer mediante porta USB. L'attuale tecnologia inizia a essere sostituita dalla cosiddetta memoria a stato solido (memoria flash) SSD.

<u>I DISCHI OTTICI</u> I dischi ottici sono supporti dove tramite un raggio laser vengono scritte o lette informazioni. La prima tecnologia prevedeva solo dischi di sola lettura, l'utente può solo leggere i dati, ma non scriverci sopra.
Poi sono arrivati i dischi sui quali è possibile scrivere una volta sola e infine i dischi sui quali si può scrivere più volte (Re-Writable). Su alcuni DVD la registrazione può avvenire in entrambe le facciate. Vengono utilizzati soprattutto per archiviare video e prodotti multimediali.

<u>**LA PEN DRIVE, MEMORIA FLASH O CHIAVETTA USB**</u> è un tipo di memoria di massa, molto utilizzata, portatile, viene collegata al computer tramite la porta USB. I dati vengono memorizzati nella cosiddetta memoria flash. L'elaboratore identifica la presenza di una chiavetta con la dicitura *disco rimovibile*. Questo tipo di periferica appartiene alla categoria plug&play, che significa collega e usa, cioè non richiede in genere istallazione e può essere collegata senza dover spegnere il computer.

<u>**LE SCHEDE SD**</u> Le memorie SD (Secure Digital) permettono di conservare informazioni in formato digitale. Oggi sia i PC, che i laptop oltre ai telefonini, alle fotocamere e videocamere, hanno appositi drive, lettori che ci consentono l'utilizzo di queste schede in lettura e scrittura, A seconda delle dimensioni abbiamo tre formati di schede: la SD (32x24 mm), la mini SD (21,5x20 mm) e la micro SD (11x15 mm).

I dispositivi interni al Case ricevono sequenze d'istruzioni dall'esterno attraverso unità periferiche di input e comunicano con l'esterno attraverso unità periferiche di output.

Tra le periferiche d'inserimento dei dati (**input**), troviamo la tastiera, il mouse, il touchpad, il touchscreen, lo scanner, il joystick, il microfono, la webcam.

Tra le periferiche di uscita dei dati (**output**), troviamo il monitor, la stampante, le cuffie, plotter, sintetizzatori vocali, ecc.

Tali periferiche possono essere collegate al computer tramite **Le porte input output**.

Esaminiamone alcune:

	✓ La porta USB permette di connettere diversi tipi di dispositivi tra cui pen drive e dischi esterni ed anche modem per la connessione a internet, garantendo velocità di trasferimento dati adeguate alle necessità.
	✓ La porta fireware è stata utilizzata per trasferire flussi multimediali, in genere da videocamere analogiche, perché più veloci degli standard delle porte usb di quel tempo.
	✓ Porta seriale, in cui i dati vengono inviati un bit alla volta. È sempre più rara trovarla nei nuovi pc in quando quasi tutte le periferiche sono prodotte con connessione USB.
	✓ La porta parallela si usa in genere per collegarci la stampante – I connettori della porta parallela hanno 25/36 pin. Anche in questo caso le periferiche che utilizzavano questo standard stanno migrando verso la tecnologia USB
	✓ Porta PS/2 serve per collegare il mouse e la tastiera.
	✓ Porta porta ethernet (LAN) che consente di collegare il computer ad una rete locale o ad un router, attraverso un cavo particolare è possibile collegare direttamente due computer

Oggi la tendenza della produzione si sta orientando verso periferiche sempre più comode ed efficienti. Poter essere svincolati da un collegamento fisico, cioè da un cavo, è una grande comodità ed è per questo che si utilizzano sempre più spesso tastiere e mouse senza fili, come anche stampanti e casse acustiche. Il canale non è più un cavo fisico ma l'etere, la trasmissione si definisce semplicemente WIRELESS (senza cavo), il segnale viene codificato e trasmesso mediante onde radio o infrarossi.

Virus e Antivirus

Un virus è un programma che può infettare file e si può riprodurre e diffondere da un sistema informatico ad un altro provocando danni più o meno gravi, che vanno dal semplice rallentamento di funzionamento fino alla cancellazione di file e al danneggiamento dell'hard disk, altri tipi di danni anche più gravi sono i furti di dati, codici di accesso e password che possono consentire addirittura transazioni finanziarie illecite a insaputa dell'utente.
Vari sono i metodi di diffusione del virus, in passato la contaminazione poteva avvenire solo caricando sul proprio computer da dischetto qualche file infetto, oggi ciò succede anche più frequentemente per il massiccio utilizzo delle pen-drive e delle reti.

I virus si diffondono anche attraverso la posta elettronica, in genere con gli allegati. Quando un utente apre l'allegato di una mail ricevuta, se questa contiene in virus, infetta il pc. Gli allegati a volte hanno frasi invitanti che possono essere di vario genere e possono arrivare anche da mittenti a noi conosciuti, ma a loro insaputa, l'apertura dell'allegato corrisponde in genere all'esecuzione del file infetto e quindi alla contaminazione.

Anche tramite il download di software da siti internet non sicuri, si può essere contaminati da virus. Per prevenire l'infezione da virus, si raccomanda di **installare un software antivirus che deve essere aggiornato con frequenza**.

Per prevenire perdite di dati, a causa di virus o per malfunzionamento del PC è consigliabile **periodicamente** creare una copia di **backup** dei dati, che deve essere fatta su supporti rimovibili come ad esempio un hard disk esterno, un cd o DVD, una chiavetta USB e tenuta in un posto sicuro.

Salute, sicurezza e ambiente

Tutti coloro che lavorano con un PC devono sapere che una postura scorretta, associata alla mancanza di pause nel lavoro, può condurre a disturbi dell'apparato muscoloscheletrico.
La postazione di lavoro deve rispondere a determinate caratteristiche, come definito nell'allegato VII del Decreto legislativo 19 settembre 1994 - n. **626**, al fine di non influire negativamente sulla salute del lavoratore.
L'Ergonomia è la disciplina scientifica che si occupa dei problemi relativi al lavoro umano integrando le ricerche e le soluzioni offerte da varie altre discipline come la medicina generale e la medicina del lavoro, la fisiologia, la psicologia, la sociologia e la fisica, al fine di realizzare un adattamento ottimale del sistema uomo-macchina-ambiente di lavoro alle capacità e ai limiti psico-fisiologici dell'uomo.

Le soluzioni ai problemi di ergonomia sono diverse per ogni singolo componente la postazione di lavoro, per esempio per la **sedia**, la migliore scelta, dal punto di vista ergonomico, è una seduta regolabile per adattare al meglio la postura, tale da poter stare seduti in posizione eretta o leggermente inclinata all'indietro, inoltre è necessario che la parte inferiore della schiena sia completamente a contatto dello schienale che deve sostenerla. Le spalle devono essere diritte, le braccia distese, vicino al corpo, con i gomiti ad angolo retto. Gli avambracci e le mani devono essere paralleli al piano di lavoro.
Il monitor, posizionato in modo scorretto può causare dolori al collo, alle spalle e affaticamento della vista a causa dei riflessi. Esso deve essere leggermente inclinato verso l'alto, la parte superiore dello schermo deve trovarsi allo stesso livello degli occhi, finestre e fonti luminose dovrebbero essere poste lateralmente al video. La distanza degli occhi varia in base alle dimensioni schermo e può essere calcolata moltiplicando la dimensione in pollici per quattro, il risultato è la distanza in centimetri, uno schermo 19 pollici richiede una distanza ottimale di circa 76 centimetri (19 x 4). La visione continuata del monitor è da evitare, circa ogni venti minuti è opportuno mettendo a fuoco un punto lontano per alcuni secondi.

Una buona norma per la tutela della salute di chi utilizza il personal computer giornalmente e per varie ore settimanali, è fare una pausa di almeno 15 minuti ogni due ore, facendo anche qualche anche esercizio fisico di stretching, è anche consigliabile di aerare regolarmente i locali specie nel caso in cui siano installati più apparecchi.

Una buona norma di sicurezza è di fare attenzione alle prese di corrente e ai fili elettrici, non inserire nella stessa presa vari cavi di alimentazione poiché potrebbe causare un sovraccarico di tensione.
Per salvaguardare l'ambiente, si consiglia di stampare solo se necessario, le nuove tecnologie permettono, infatti, di trasferire e far circolare facilmente le informazioni in formato elettronico, inoltre si raccomanda di riciclare e smaltire correttamente le apparecchiature e i materiali utilizzati.

2. Codifica dell'informazione

Autori: **Alessandra D'Orazio, Giorgia Martina, Paolo Lillo**
Rielaborazione e riduzione a cura di: **Angelo Oliva, Dario Rinaudo**

Competenze	Abilità	Conoscenze
Sapere codificate un'informazione numerica *Saper codificare un'informazione alfanumerica* *Saper individuare le tecniche di codifica conformi ai dati coinvolti*	*Riconoscere i diversi tipi di informazioni*	*Dato e informazione* *Informazione Analogica e digitale* *Bib e byte* *Codifica dei dati*

La codifica dell'informazione, il bit

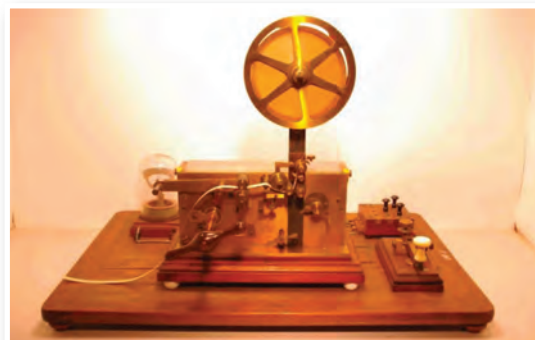
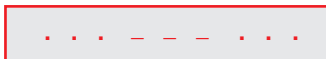
Diamo una definizione di "INFORMATICA":

> **"L'informatica è la disciplina che si occupa della *codifica*, della memorizzazione, della trasmissione e dell'elaborazione dell'INFORMAZIONE"**

Intendiamo per INFORMAZIONE
"Ogni messaggio o elemento conoscitivo che è possibile comunicare o acquisire".

La parola e le espressioni del corpo, i segni (dai graffiti preistorici alla scrittura) e in generale tutto ciò che può essere percepito dai sensi, hanno permesso all'uomo nel corso della storia di comunicare, cioè di trasferire l'informazione non solo da uomo a uomo ma anche da generazione a generazione. I modelli culturali di oggi sono il risultato della lunga evoluzione del processo.

Se la comunicazione è alla base del trasferimento delle informazioni, il ragionamento e quindi l'uso della capacità logica-deduttiva hanno da sempre avuto il compito di elaborare l'informazione attraverso l'uso di modelli interpretativi che con lo scorrere del tempo sono divenuti sempre più articolati e raffinati.
L'uomo, del resto, si è sempre ingegnato per superare i suoi limiti utilizzando tecniche e strumenti per rendere sempre più efficace la sua capacità di comunicare: dall'uso dei tam-tam e dei segnali di fumo alle moderne tecnologie.

Il lettore avrà certamente sentito parlare del "codice Morse": quando nel 1838 si realizzarono i primi telegrafi via cavo (e successivamente via radio) gli unici "segnali" che era possibile produrre e distinguere in modo chiaro e immediatamente comprensibile erano due segnali acustici di lunghezza lunga e breve, rispettivamente linea e punto; la produzione dei segnali avveniva da un lato della linea telegrafica (trasmettitore), agendo su un interruttore a molla: ad una pressione breve corrispondeva un segnale del tipo punto mentre ad una pressione lunga faceva seguito un segnale del tipo linea; dall'altro lato della linea telegrafica il segnale elettrico ricevuto attivava un elettromagnete, il cui movimento portava in contatto una penna su una striscia di carta trascinata da un meccanismo a orologeria.

Il messaggio più famoso del codice Morse è certamente SOS, il segnale di richiesta di soccorso (“Save Our Souls” cioè “Salvate le nostre anime”) corrispondente alla sequenza: · · · – – – · · ·

dove i primi tre punti stanno per “S”, le tre linee stanno per “O” e gli ultimi tre punti stanno nuovamente per “S”. Punto e linea rappresentano quindi i 2 simboli grazie ai quali è possibile rappresentare i messaggi dell'alfabeto Morse.

Quindi, una volta stabilita la corrispondenza tra l'insieme di punti e linee e le lettere dell'alfabeto, le cifre, i caratteri di interpunzione e alcuni segni convenzionali (ovvero una volta stabilita la **codifica** nell'alfabeto Morse di ogni singolo carattere che compone un testo scritto), un qualsiasi brano può essere codificato utilizzando punti e linee; d'altro canto, chiunque abbia a disposizione una tabella di conversione

punti-linee → carattere

sarà in grado di leggere il testo originale.

A	· –	J	· – – –	S	· · ·
B	– · · ·	K	– · –	T	–
C	– · – ·	L	· – · ·	U	· · –
D	– · ·	M	– –	V	· · · –
E	·	N	– ·	W	· – –
F	· · – ·	O	– – –	X	– · · –
G	– – ·	P	· – – ·	Y	– · – –
H	· · · ·	Q	– – · –	Z	– – · ·
I	· ·	R	· – ·		

L'alfabeto Morse è un esempio di CODIFICA; è chiaro che per la rappresentazione codificata dell'informazione è necessaria una CONVENZIONE (detta anche PROTOCOLLO), cioè un accordo tra le parti che devono comunicare sul SIGNIFICATO da dare alle sequenze di simboli: tanto più diffusa è tale convenzione tanto più estesa sarà la platea dei possibili fruitori.
L'introduzione dell'informatica ha avviato un'epoca in cui l'informazione e il trattamento dell'informazione, grazie all'evoluzione della tecnologia elettronica, hanno assunto forme sempre più automatiche. Al fine di potere essere “gestita” da processi automatici è stato indispensabile rendere l'informazione sempre più autonoma cioè indipendente dalla capacità interpretativa dell'uomo.
Prima di entrare nel vivo della trattazione sulla codifica dei più comuni tipi di informazione (spiegheremo tra un po' cosa intendiamo per “tipi”), soffermiamo la nostra attenzione sulla definizione e sulla codifica di un'informazione in ambiente più specificatamente informatico.
Prima domanda: se l'informazione è l'oggetto intorno a cui ruota la disciplina Informatica, che cosa è un dato e che relazione intercorre tra un dato e un'informazione?
Seconda domanda: nel campo dell'informatica, come si può memorizzare un'informazione e in quanti modi diversi è possibile rappresentare la stessa informazione?

Rispondiamo alla prima domanda: i termini **dato** e **informazione** sono a volte utilizzati come sinonimi, ma i due concetti non coincidono. Facciamo alcuni esempi:
- il numero intero 16: è un dato o un'informazione? È un dato, diventa un'informazione se aggiungo che 16 è l'età di una ragazza oppure che 16 sono gli amici che ho invitato alla mia festa.
- Maria ha i capelli rossi: è un dato o un'informazione? È un'informazione: il dato, colore rosso, è “diventato” un'informazione in quanto associato ai capelli di Maria.

Quindi: un dato è una parte dell'informazione, la quale a sua volta può essere considerata come un dato a cui è associato il significato che ha nel contesto di studio. Più precisamente, possiamo considerare l'informazione costituita da tre elementi:
1. il valore (il dato)
2. il tipo (carattere, numero intero, stringa,..) utilizzato per esprimere il valore
3. il significato (la semantica) da associare al valore

Se l'informatica si occupa, tra le varie cose, così come detto nella definizione iniziale, di memorizzare ed elaborare le informazioni, che cosa verrà trascritto ovvero memorizzato e che cosa verrà elaborato? La risposta è: **i dati**!

Possiamo ora rispondere alla seconda domanda, iniziando con il costatare che i dati sono memorizzati nelle memorie (di massa, centrale, RAM, ROM, ..) ed elaborati dai computer:
memorie e computer sono supporti fisici costituiti da dispositivi in grado di distinguere tra due (e solo due) diversi valori di riferimento di una grandezza fisica (assenza/presenza di tensione, luminoso/non luminoso). Tali dispositivi sono detti elettricamente bistabili.

Se associamo a ciascun valore un simbolo, possiamo rappresentare i due diversi stati del dispositivo in esame con due simboli distinti; i due simboli utilizzati sono {0, 1}, che costituiscono l'alfabeto binario. Attenzione: 0 e 1 non sono numeri ma simboli, da non confondere quindi con i primi due numeri naturali.

Il Bit: possiamo ora definire il concetto informatico di bit (**Bi**nary Digi**t** = cifra binaria)

"Il bit è l'unità elementare di informazione cioè la minima quantità di informazione codificabile"

Essendo indispensabili, come appena visto, almeno due simboli per rappresentare un'informazione, potremmo ridefinire il bit come **uno tra i DUE simboli** possibili adottati nello specifico contesto binario ('punto' e 'linea' nell'alfabeto Morse oppure '0' e '1' nel sistema numerico binario). L'uso di un alfabeto binario (cioè basato sull'adozione di un alfabeto di due simboli) è largamente utilizzato nella tecnologia in virtù della facile e quindi economica riproducibilità nei vari contesti della fisica (elettricità, magnetismo, ottica) di **uno tra due stati fisici**: tensione/nontensione (elettricità), magnetizzato/smagnetizzato (magnetismo), luminoso/non luminoso (ottica).

Il bit vale 0 se è NO, FALSO, SPENTO, NON PASSA CORRENTE.

Il bit vale 1 se è SI', VERO, ACCESO, PASSA CORRENTE.

Il bit è quindi il più piccolo dato memorizzabile in un elaboratore e può rappresentare una sola informazione binaria. Qualsiasi altro tipo di informazione è rappresentato come sequenze finite di 0 e 1; come questi 2 simboli sono combinati tra loro dipende dal codice usato, ovvero dalla relazione che trasforma ogni dato in una stringa di bit.
Ma una sequenza di bit cosa rappresenta? A quale possibile informazione è associata?
Nei paragrafi successivi daremo una risposta a questi quesiti; più precisamente affronteremo il problema di come è possibile codificare:

- numeri
- caratteri
- immagini
- suoni
- video

Codifica dei numeri

Iniziamo con il ricordare alcune definizioni:

- **Numero**: oggetto o ente astratto
- **Numerale**: insieme di simboli che rappresenta un numero in un dato sistema di numerazione
- **Sistema di numerazione**: un insieme di simboli e un insieme di regole; i simboli servono per rappresentare un numero, le regole stabiliscono come scrivere e come operare con i numerali.

Di conseguenza possiamo rappresentare un numero e scrivere il suo numerale solo dopo aver stabilito il sistema di numerazione che intendiamo utilizzare: se cambiamo sistema di numerazione, cambia la rappresentazione del numero.

Esempio: il numerale 15 nel sistema decimale diventa XV in numeri romani.

Esaminiamo allora due caratteristiche del sistema di numerazione che usiamo correntemente, il sistema decimale:

- è in base 10: ogni numero è rappresentato da una combinazione di 10 simboli diversi {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
- è posizionale: esiste una regola che stabilisce un legame tra il simbolo e la sua posizione nel numerale che rappresenta il numero, cioè ogni cifra assume un valore diverso a seconda della posizione occupata nella rappresentazione del numero.

  Esempio: numerale = 184 → $1*10^2 + 8* 10^1 + 4*10^0$ = 100 + 80 +4
  posizione → 210
  (N.B. Il simbolo di moltiplicazione è indicato con "" e non con "x", in conformità con la simbologia adottata in ambiente informatico)*

Come si può osservare, la posizione delle cifre in un numerale è tanto più importante, perché è quella che più contribuisce al valore finale (in gergo "più significativa"), quanto più a sinistra si trova nella sequenza numerica: la cifra MENO SIGNIFICATIVA è quella più a destra, che nel sistema decimale rappresenta le "unità".

Il sistema decimale deve ad una proprietà anatomica (le 10 dita delle mani) la sua diffusione, ma non risulta altrettanto vantaggioso quando l'elaborazione dei numeri si sposta dall'uomo al computer.
Come già accennato nel paragrafo precedente, un computer in quanto macchina “sente” 2 diversi stati (acceso/spento, presenza/assenza di tensione); conseguenza quasi ovvia: per codificare i numeri che devono essere elaborati da un computer si utilizza il sistema di numerazione in base 2, ovvero il sistema di numerazione posizionale che ha il bit come elemento di base.
Prenderemo in esame altri 2 sistemi di numerazione posizionali, il sistema di numerazione in base 8 (ottale) e il sistema in base 16 (esadecimale), entrambi caratterizzati dal fatto che le basi sono potenze della base 2 (spiegheremo in seguito cosa comporta questa considerazione).

Due annotazioni:
- useremo il termine numero anche quando sarebbe più corretto utilizzare numerale
- ogni sequenza di simboli che scriveremo avrà come pedice la base in cui è espresso il numero.

Esempio: 215_{10} rappresenta un numero espresso in base 10
1011_2 rappresenta un numero espresso in base 2 (si legge **uno zero uno uno**)

Sistema di numerazione in base 2

Il sistema binario utilizza due simboli, 0 e 1, grazie ai quali possiamo rappresentare qualsiasi numero, purché si abbia a disposizione un numero di bit sufficienti. Osserviamo la seguente una tabella in cui riportiamo tutti i valori rappresentabili con un determinato numero di bit

n° di bit	numeri rappresentabili															
1	0	1														
2	00	01	10	11												
3	000	001	010	011	100	101	110	111								
4	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

In generale con n bit si possono rappresentare interi positivi compresi nell'intervallo $[0, 2^n-1]$.
Attenzione: una volta che è stato fissato n, occorre rappresentare anche gli 0 non significativi.

Sistema di numerazione in base 8

- è un sistema di numerazione posizionale
- ogni numero è espresso come combinazione degli 8 simboli: 0, 1, 2, 3, 4, 5, 6, 7

Sistema di numerazione in base 16

- è un sistema di numerazione posizionale
- ogni numero è espresso come combinazione dei 16 simboli:
  0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

Scriviamo i primi 21 numeri naturali in base 10, in base 2, in base 8, in base 16 (per semplicità, non ci preoccupiamo della lunghezza delle rappresentazioni nelle varie basi).

base 10	base 2	base 8	base 16
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10
17	10001	21	11
18	10010	22	12
19	10011	23	13
20	10100	24	14

Ma allora: se non abbiamo a disposizione la tabella precedente, come possiamo affermare che la sequenza 1101 in base 2 equivale a 13 in base 10 (1101_2=13_{10})?

E soprattutto: come procediamo per valori maggiori di 20? 34710 in base 10 a quale valore corrisponde in base 2? e in base 16? e in base M?

Esiste una tecnica, semplice e intuitiva, che permette di convertire un numero espresso in base 10 nell'equivalente in un'altra base M (in effetti è una tecnica che permette la conversione da una qualsiasi base N a una qualsiasi altra base M). Ricordiamo che stiamo trattando numeri naturali, quindi senza parte decimale e senza segno.

La regola di conversione di un numero X in base 10, X_{10}, nella base M è la seguente:
1. si esprime M in base 10 (indichiamo il numero ottenuto con M_{10})
2. si procede per divisioni successive, dove il divisore è sempre M_{10}, il dividendo, invece, è X_{10} la prima volta, successivamente diventa il quoziente ottenuto nella divisione precedente
3. si termina il procedimento quando il valore del quoziente è 0
4. si prendono i resti delle divisioni effettuate in ordine inverso, dall'ultimo al primo
5. si convertono, per basi M>10, i resti ottenuti, espressi in base 10, nella corrispondente cifra in base M.

Rileggiamo quanto esposto con M=2.

Conversione da base 10 a base 2
1. (superfluo) si esprime 2 in base 10
2. si procede per divisioni successive del numero dato per 2
3. si termina il procedimento quando il valore del quoziente è 0
4. i resti delle singole divisioni vengono presi in ordine inverso rispetto a quello del calcolo e rappresentano le cifre del numero espresso in base 2 (ricordati che i soli possibili resti saranno 0 o 1).
5. (superfluo)

Esempio: convertire 106_{10} in base 2.

106:2 =53 resto 0
53:2 =26 resto 1
26:2 =13 resto 0
13:2 =6 resto 1
6:2 =3 resto 0
3:2 =1 resto 1
1:2 =0 resto 1

6	53	26	13	6	3	1	0	← quozienti
	1	0	1	0	1	1		← resti

quindi 106_{10}=1101010_2

Proponiamo altri esempi con base M diversa da 2:
• convertire 106_{10} in base 5

106:5 = 21 resto 1
21:5 = 4 resto 1
4:5 = 0 resto 4

106	21	4	0	← quozienti
1	1	4		← resti

otteniamo: $(106)_{10}$ = $(411)_5$

• convertire 180_{10} in base 16

180:16 = 11 resto 4
11:16 = 0 resto 11

180	11	0	← quozienti
4	11		← resti

Ricorda che il resto 11 corrisponde alla B del codice esadecimale quindi otteniamo $(180)_{10}$ = $(B4)_{16}$

Consideriamo ora le conversioni da base M a base 10. Enunciamo la regola che permette di convertire un numero espresso in una base qualsiasi nella corrispondente rappresentazione decimale (ricorda che i sistemi coinvolti sono tutti sistemi posizionali)

1. si moltiplica ogni cifra per il valore della base elevato ad esponente pari alla posizione che ha la cifra nella sequenza di simboli
2. si sommano i valori così ottenuti

Rileggiamo quanto esposto con M=2.

Conversione da base 2 a base 10

1. si moltiplica ogni *bit* per il valore della base elevato ad esponente pari alla posizione che ha il *bit* nella sequenza di simboli
2. si sommano i valori così ottenuti.

Esempio: convertire $(11011)_2$ in base 10.

1 1 0 1 1= $1*2^4 + 1*2^3 + 0*2^2 + 1*2^1 + 1*2^0$ = 16 + 8 +0 + 1 + 1 = 27 in base 10

posizione → 4 3 2 1 0

Nell'esempio precedente, la cifra meno significativa è il coefficiente della potenza 2^0, la cifra più significativa (1 in posizione 4) è il coefficiente di 2^4.

Proponiamo altri esempi con base M diversa da 2:

Conversione dalla base M alla base 10

• convertire 214_6 in base 10 ricordando che la cifra meno significativa indica il coefficiente di 6^0 quella più significativa il coefficiente di 6^2, segue

$214_6 = 2* 6^2 + 1* 6^1 + 4* 6^0 = 2* 36 + 1*6 + 4*1 = 72 + 6 + 4= 82_{10}$

Conversione da base dirette per basi (2, 8 ,16)

• Nel caso generale, con N e M valori qualsiasi, è possibile applicare la regola espressa inizialmente ma, spesso, si preferisce trasformare il numero dalla base N alla base 10 e poi far seguire la conversione da base 10 a base M.

Esempio: 45_7 = $numero_3$

1° passo: si converte 45_7 in base 10 → $45_7 = 4*7^1 + 5* 7^0 = 28 + 5 = 33_{10}$

2° passo: si converte il numero così ottenuto in base 3 → $33_{10} = 1020_3$

• Più semplice è la conversione se la base N è 2 e la base M è 8 oppure 16 (ma vale anche il viceversa): l'identità $8 = 2^3$ permette di convertire ogni gruppo di 3 bit in un simbolo in base 8 e, viceversa, ogni simbolo in base 8 nella sequenza di 3 bit in base 2. Nel caso in cui la base è 16 ($16 = 2^4$), si ripete lo stesso procedimento ma con gruppi di 4 (e non 3) bit.

Gli esempi chiariranno questo procedimento.

Esempio: convertire 1110010101000100_2 in ottale

Partendo dalla cifra meno significativa, si considerano le cifre binarie rispettivamente a gruppi di 3:

001	110	010	101	000	100
↓	↓	↓	↓	↓	↓
1	6	2	5	0	4

Quindi: $1110010101000100_2 = 162504_8$

Esempio: convertire $11100111010100 1100_2$ in esadecimale Partendo dalla cifra meno significativa, si considerano le cifre binarie rispettivamente a gruppi di 4:

0011	1001	1101	0100	1100
↓	↓	↓	↓	↓
3	9	D	4	C

Quindi: $111001110101001100_2 = 39D4C_{16}$

Esempio: convertire 372_8 in binario

Ogni cifra viene codificata nel valore corrispondente in base 2 espresso con 3 bit:

3	7	2
↓	↓	↓
011	111	010

Quindi: $372_8 = 011111010_2$

Esempio: convertire $C2E3_{16}$ in binario
Ogni cifra viene codificata nel valore corrispondente in base 2 espresso con 4 bit:

C	2	E	3
↓	↓	↓	↓
1100	0010	1110	0011

Quindi: $C2E3_{16} = 1100001011100011_2$

Esercizi: conversioni di base

a) *da base 10 a base 2*

1) 23 [10111]
2) 38 [100110]
3) 100 [1100100]
4) 32 [100000]
5) 70 [1000110]
6) 65 [1000001]
7) 98 [1100010]
8) 255 [11111111]
9) 318 [100111110]
10) 512 [1000000000]

b) *da base 10 alla base indicata*

11) 34 in base 4 [202_4]
12) 128 in base 6 [332_6]
13) 7 in base 7 [10_7]
14) 100 in base 16 [64_{16}]
15) 1024 in base 5 [1014_9]
16) 38 in base 3 [1102_3]
17) 42 in base 7 [60_7]
18) 78 in base 3 [2220_3]
19) 153 in base 11 [$12A_{11}$]

c) *da base 2 a base 10*

20) 1011 [11]
21) 10110 [22]
22) 110011 [51]
23) 111111 [63]
24) 1011100 [92]
25) 11000000 [192]
26) 111010101 [469]
27) 100000000 [256]
28) 10000011 [131]
29) 110011001 [409]

d) da base N a base 10

30) 152_8	[106]
31) 123_4	[27]
32) 342_5	[97]
33) $A1F_{16}$	[2591]
34) 351_7	[183]
35) 1234_4	[*impossibile*]
36) 100_5	[25]
37) 100_6	[36]
38) 101_7	[50]
39) $A19_{12}$	[1461]

e) da base N a base M

40) 134_6 in base 2	[111010_2]
41) 615_8 in base 2	[110001101_2]
42) 326_9 in base 16	[$10B_{16}$]
43) 714_8 in base 7	[1225_7]
44) 1110110_2 in base 5	[433_5]
45) 171_8 in base 16	[79_{16}]
46) 1110110_2 in base 16	[76_{16}]
47) 11110110_2 in base 8	[366_8]
48) $1AB6_{16}$ in base 2	[1101010110110_2]
49) ABC_{16} in base 8	[5274_8]

Codifica dei caratteri

I numeri rappresentano solo una piccola parte delle informazioni memorizzate ed elaborate da un computer; insieme alla codifica delle informazioni numeriche è sorto il problema della rappresentazione dei dati alfanumerici, ovvero dei documenti testuali e anche in questo caso sono state elaborate leggi di trasformazione da carattere a stringa di bit. Prima però di entrare nel dettaglio è fondamentale introdurre la definizione di byte.

Si definisce BYTE una sequenza di 8 BIT (evidente anche in questo caso la naturale predilezione che si ha in informatica per le potenze del 2 come conseguenza dell'uso della codifica binaria). Essendo codificabili su 8 bit 256 combinazioni distinte di 0/1, si ritenne che tale quantità di bit fosse sufficiente per rappresentare tutti i simboli utilizzabili nella scrittura di messaggi e documenti e da allora il BYTE è divenuta una sorta di unità di misura della codifica (1 BYTE = 1 carattere), tuttora utilizzata per definire la capacità di memoria: sempre privilegiando le potenze del 2 e utilizzando il simbolo “B” per riferirsi al byte (ricorda che “b” è usato per indicare grandezze espresse in bit), si definisce:

sigla	descrizione	Quantità in byte
1kB	1 kilo-byte	1024 bytes
1MB	1 mega-byte	1024^2 bytes = 1.048.576 bytes
1GB	1 giga-byte	1024^3 bytes = 1.073.741.824 bytes
1TB	1 tera-byte	1024^4 bytes = 1.099.511.627.776 bytes

Una unità a disco da 1TB (un tera-byte) è dunque in grado di memorizzare più di mille miliardi di caratteri.

Si è detto che su 1 byte cioè su 8 bit è possibile codificare un carattere: come stabilire allora la corrispondenza tra byte e carattere? La corrispondenza non può che essere basata su una “convenzione”, cioè su un accordo stabilito da un'apposita commissione composta da tecnici, studiosi, rappresentanti di aziende, istituzioni e governi che scelgono uno standard sul quale uniformarsi, i due standard più diffusi e noti sono il codice ASCII e il codice UNICODE.

Il codice **ASCII** (**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange ovvero *codifica standard americana per lo scambio di informazioni*) è nato come codice a 7 bit: sono sufficienti infatti 7 bit (quindi 128 combinazioni di 0/1) per rappresentare tutti i caratteri maiuscoli e minuscoli dell'alfabeto inglese, le cifre da 0 a 9, i simboli di punteggiatura, le parentesi, gli operatori matematici (“+”, “-”, “*”, “\”, “%”), gli operatori relazionali (“=”, “>”, “<”) oltre ad alcuni caratteri “non stampabili”, cioè non corrispondenti a simboli, ma testuali utilizzati originariamente come comandi nelle trasmissioni telegrafiche. All'inizio degli anni 1960 con l'affermarsi del byte anche come unità di misura delle memorie, fu aggiunto un bit, passando così dalla codifica ASCII alla codifica ASCII estesa; l'aggiunta di un bit ha permesso di raddoppiare il numero di caratteri codificabili (da 128 a 256), permettendo la codifica dei caratteri nazionali (per esempio le vocali accentate: ora è possibile codificare età e non *eta*').
L'estendersi delle comunicazioni tra paesi che utilizzano alfabeti diversi dall'alfabeto inglese ha comportato la necessità di ampliare ulteriormente l'insieme dei simboli da codificare e, conseguentemente, è stato giudicato inadeguato un solo byte per carattere; da qui il diffondersi del codice UNICODE, che associa ad ogni carattere 16 bit, cioè 2 byte e che quindi permette la codifica di $2^{16} = 65536$ simboli diversi.

Segue la tabella della codifica ASCII standard su 8 bit (i primi 128 caratteri)

Dec	Hx	Oct	Char	Dec	Hx	Oct	Chr	Dec	Hx	Oct	Chr	Dec	Hx	Oct	Chr
0	0	000	NUL (null)	32	20	040	Space	64	40	100	@	96	60	140	`
1	1	001	SOH (start of heading)	33	21	041	!	65	41	101	A	97	61	141	a
2	2	002	STX (start of text)	34	22	042	"	66	42	102	B	98	62	142	b
3	3	003	ETX (end of text)	35	23	043	#	67	43	103	C	99	63	143	c
4	4	004	EOT (end of transmission)	36	24	044	$	68	44	104	D	100	64	144	d
5	5	005	ENQ (enquiry)	37	25	045	%	69	45	105	E	101	65	145	e
6	6	006	ACK (acknowledge)	38	26	046	&	70	46	106	F	102	66	146	f
7	7	007	BEL (bell)	39	27	047	'	71	47	107	G	103	67	147	g
8	8	010	BS (backspace)	40	28	050	(	72	48	110	H	104	68	150	h
9	9	011	TAB (horizontal tab)	41	29	051	)	73	49	111	I	105	69	151	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	74	4A	112	J	106	6A	152	j
11	B	013	VT (vertical tab)	43	2B	053	+	75	4B	113	K	107	6B	153	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	76	4C	114	L	108	6C	154	l
13	D	015	CR (carriage return)	45	2D	055	-	77	4D	115	M	109	6D	155	m
14	E	016	SO (shift out)	46	2E	056	.	78	4E	116	N	110	6E	156	n
15	F	017	SI (shift in)	47	2F	057	/	79	4F	117	O	111	6F	157	o
16	10	020	DLE (data link escape)	48	30	060	0	80	50	120	P	112	70	160	p
17	11	021	DC1 (device control 1)	49	31	061	1	81	51	121	Q	113	71	161	q
18	12	022	DC2 (device control 2)	50	32	062	2	82	52	122	R	114	72	162	r
19	13	023	DC3 (device control 3)	51	33	063	3	83	53	123	S	115	73	163	s
20	14	024	DC4 (device control 4)	52	34	064	4	84	54	124	T	116	74	164	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	85	55	125	U	117	75	165	u
22	16	026	SYN (synchronous idle)	54	36	066	6	86	56	126	V	118	76	166	v
23	17	027	ETB (end of trans. block)	55	37	067	7	87	57	127	W	119	77	167	w
24	18	030	CAN (cancel)	56	38	070	8	88	58	130	X	120	78	170	x
25	19	031	EM (end of medium)	57	39	071	9	89	59	131	Y	121	79	171	y
26	1A	032	SUB (substitute)	58	3A	072	:	90	5A	132	Z	122	7A	172	z
27	1B	033	ESC (escape)	59	3B	073	;	91	5B	133	[	123	7B	173	{
28	1C	034	FS (file separator)	60	3C	074	<	92	5C	134	\	124	7C	174	|
29	1D	035	GS (group separator)	61	3D	075	=	93	5D	135	]	125	7D	175	}
30	1E	036	RS (record separator)	62	3E	076	>	94	5E	136	^	126	7E	176	~
31	1F	037	US (unit separator)	63	3F	077	?	95	5F	137	_	127	7F	177	DEL

Riportiamo per completezza anche i successivi 128 caratteri della codifica ASCII estesa (solo la codifica decimale)

128	Ç	144	É	160	á	176	░	193	┴	209	╤	225	ß	241	±
129	ü	145	æ	161	í	177	▒	194	┬	210	╥	226	Γ	242	≥
130	é	146	Æ	162	ó	178	▓	195	├	211	╙	227	π	243	≤
131	â	147	ô	163	ú	179	│	196	─	212	╘	228	Σ	244	⌠
132	ä	148	ö	164	ñ	180	┤	197	┼	213	╒	229	σ	245	⌡
133	à	149	ò	165	Ñ	181	╡	198	╞	214	╓	230	µ	246	÷
134	å	150	û	166	ª	182	╢	199	╟	215	╫	231	τ	247	≈
135	ç	151	ù	167	º	183	╖	200	╚	216	╪	232	Φ	248	°
136	ê	152	_	168	¿	184	╕	201	╔	217	┘	233	Θ	249	·
137	ë	153	Ö	169	_	185	╣	202	╩	218	┌	234	Ω	250	
138	è	154	Ü	170	¬	186	║	203	╦	219	█	235	δ	251	√
139	ï	156	£	171	½	187	╗	204	╠	220	▄	236	∞	252	_
140	î	157	¥	172	¼	188	╝	205	═	221	▌	237	φ	253	²
141	ì	158	_	173	¡	189	╜	206	╬	222	▐	238	ε	254	■
142	Ä	159	ƒ	174	«	190	╛	207	╧	223	▀	239	∩	255	
143	Å	192	└	175	»	191	┐	208	╨	224	α	240	≡		

A titolo di esempio, la sequenza ASCII di codifica del testo "*Ciao, mondo*!":

C	i	a	o	,		m	o	n	d	o	!
in esadecimale											
43	69	61	6F	2C	20	6D	6F	6E	64	6F	21
in binario											
01000011	01101001	01100001	01101111	00101100	00010000	01101101	01101111	01101110	01100100	01101111	00100001

E' piuttosto evidente che, se dovessimo codificare in binario un documento senza l'aiuto di un qualche automatismo, ci troveremmo in grave difficoltà così come nella stessa difficoltà si troverebbe chi dovesse decodificarlo nella necessità di ricostruire il documento originale

"Ciao, mondo!" > codificatore > sequenza di bit

sequenza di bit > decodificatore > "Ciao, mondo!"

Nel precedente schema il "codificatore" e il "decodificatore" sono degli strumenti che, conoscendo perfettamente il criterio (cioè le regole) di codifica, sono in grado di effettuare tanto la trasformazione da documento testuale a documento binario quanto la trasformazione inversa.

Codifica delle immagini (fisse)

Così come un testo può essere memorizzato su un supporto elettronico, dopo essere stato codificato in una sequenza di bit grazie a codici come ASCII o UNICODE, anche le immagini, i suoni e i video possono essere memorizzati purché opportunamente "trasformati" in sequenze di 0 e 1. A differenza, però, di un documento testuale, un'immagine non nasce come un oggetto discreto, ovvero costituito da un insieme finito e ben determinato di simboli (caratteri dell'alfabeto inglese maiuscoli e minuscoli, cifre, simboli aritmetici, segni di punteggiatura, etc.), ma è un insieme di colori, linee, forme, sfumature che combinate tra loro rappresentano appunto un'immagine (una fotografia è un'immagine, un quadro è un'immagine, la stampa di un carattere è un'immagine). Occorre quindi trasformare l'immagine in una opportuna sequenza di simboli che, a loro volta, verranno codificati in sequenze di 0 e 1. Più precisamente occorre DIGITALIZZARE l'immagine, cioè passare dalla sua rappresentazione analogica alla corrispondente rappresentazione digitale.

Ma cosa intendiamo in generale con i termini analogico e digitale?

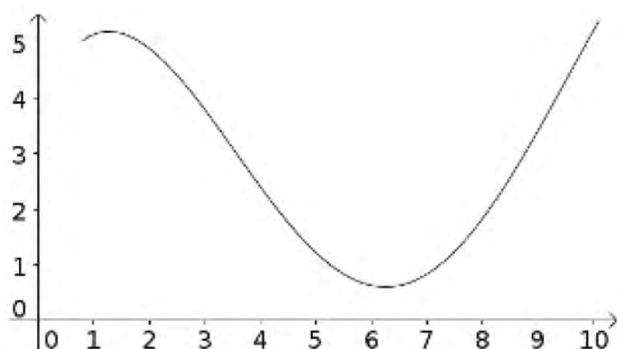
Una grandezza si definisce **analogica** quando varia in modo continuo nel tempo e/o nello spazio e nei valori assunti, cioè quando può assumere qualsiasi valore in un determinato intervallo. Una grandezza si definisce **digitale** quando NON varia in modo continuo: essa può assumere valori che sono elementi di un insieme discreto e quindi può assumere in un determinato intervallo solo alcuni valori e non altri (si dice che una grandezza digitale assume valori in modo discontinuo o a salti).

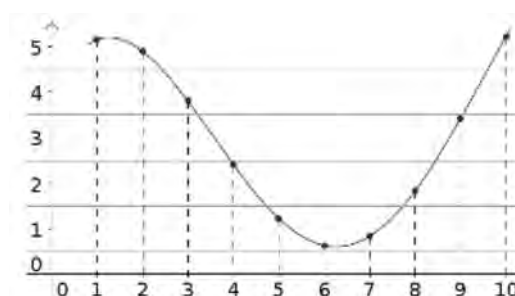
Consideriamo due grandezze fisiche con cui interagiamo quotidianamente, tempo e temperatura: siamo abituati a rappresentare il loro valore sia in "analogico" che in "digitale".

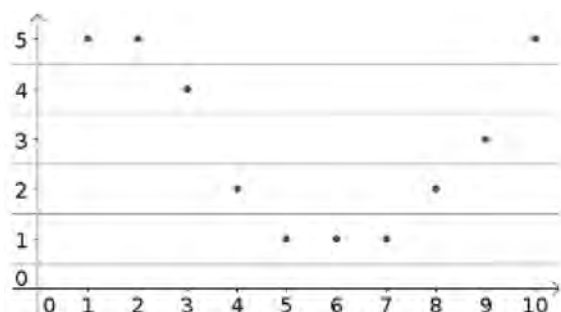
Grandezza (e relativa unità di misura)	Misura analogica	Misura digitale
Tempo (ora, minuti)		
temperatura (gradi)		

Torniamo al termine digitalizzare con una definizione: “Nel campo dell'informatica e dell'elettronica, con digitalizzazione si intende il processo di trasformazione di un'immagine, di un suono, di un documento in un formato digitale, interpretabile da un computer, dove per formato digitale si intende un codice binario in cui tutto è rappresentato da combinazioni di zero od uno, quindi da stati del tipo acceso/spento.” (Wikipedia)

il processo di digitalizzazione è composto da una sequenza di tre fasi:

1. il campionamento.
2. la quantizzazione.
3. la codifica.

Considerare un segnale analogico qualsiasi: esso può essere rappresentato graficamente tramite una “linea” come nella figura a fianco.

Possiamo, ad esempio, pensare che sull'asse delle ascisse rappresentiamo il tempo e sull'asse delle ordinate il valore che il segnale assume in quell'istante. I valori cambiano con continuità (la funzione è continua) nell'intervallo di tempo considerato.

Eseguiamo i passi necessari per “digitalizzare” questa grandezza.

Campionamento: i valori della grandezza vengono rilevati ad intervalli di tempo distinti (la grandezza è ancora analogica).

Quantizzazione: si definisce l'intervallo di valori all'interno del quale ogni singolo valore campionato viene approssimato (in questa fase si introduce un errore che è inversamente proporzionale al numero di valori che vengono fissati).

Codifica: ad ogni valore che la grandezza assume nell'intervallo di campionamento viene associato un numero (una sequenza di 0 e 1) in base ad un codice.

Campione	Valore	CODIFICA			
1	5	0	1	0	1
2	5	0	1	0	1
3	4	0	1	0	0
4	2	0	0	1	0
5	1	0	0	0	1
6	1	0	0	0	1
7	1	0	0	0	1
8	2	0	0	1	0
9	3	0	0	1	1
10	5	0	1	0	1

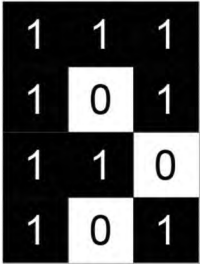
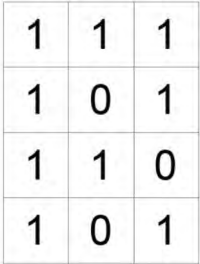
Dopo questa digressione, torniamo all'argomento del paragrafo, la digitalizzazione delle immagini: descriveremo due strategie, la grafica bitmap o raster e la grafica vettoriale.

Grafica bitmap o raster.

Consiste nel sovrapporre all'immagine una griglia di punti o, meglio, di suddividere l'immagine in una serie di celle di dimensioni ridotte (quanto ridotte lo vedremo tra poco): ogni punto o cella prende il nome di pixel (PICture ELement); il pixel è quindi il più piccolo elemento costituente un'immagine digitalizzata. Ma in quanti pixel possiamo suddividere un'immagine? Non esiste un valore fisso, esiste però il concetto di “fedeltà” all'immagine iniziale o, meglio, di “qualità” dell'immagine riprodotta. Vediamo di spiegare questo concetto considerando, come esempio, le seguenti immagini:

Larghezza 25 pixel (*) Altezza 27 pixel	Larghezza 64 pixel (*) Altezza 67 pixel	Larghezza 256 pixel Altezza 270 pixel

(*) valori approssimati
(N.B. tutte e tre le immagini hanno la stessa dimensione)

Possiamo quindi affermare che all'aumentare del numero di pixel, mantenendo costanti le dimensioni della figura, riusciamo ad ottenere un'immagine “migliore” cioè sempre più fedele all'originale.
Il numero di pixel in cui si suddivide lo schermo prende il nome di **definizione**, a cui è strettamente connesso, ma non è un sinonimo, il concetto di **risoluzione**, che coincide con la quantità di pixel per unità di misura. La risoluzione è quindi la densità di pixel e da essa dipende la qualità dell'immagine; nell'esempio in figura abbiamo assunto come unità di misura la dimensione dell'immagine e modificato la quantità di pixel in essa contenuti: aumentando la densità è aumentata la qualità. L'unità di misura più frequentemente usata per quantificare la risoluzione è il **dpi** (Dot Per Inches), numero di pixel per pollice (un pollice= 2,54 cm).

Esiste un terzo parametro che caratterizza un'immagine digitalizzata: la **profondità** dell'immagine (o del colore), anch'essa rappresentata con un certo numero di bit: maggiore è il numero di bit utilizzati maggiore è la qualità del colore riprodotto.

Infatti un'immagine è fatta anche e soprattutto di colori, quindi ad un pixel occorre associare un numero che codifica il colore che ha, nell'immagine da riprodurre, il punto rappresentato.
Per semplicità supponiamo di voler rappresentare un carattere, per esempio il carattere 'R', in bianco e nero: basterà un solo bit per rappresentare il colore, per esempio 1 per il nero, 0 per il bianco.

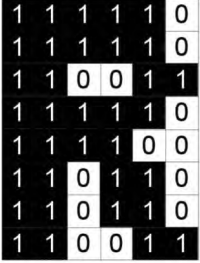
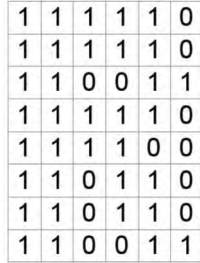



Immagine del carattere 'R'	Immagine con sovrapposta una griglia di 4x3 pixel

(le due figure successive sono state ingrandite per permettere una maggiore leggibilità)

Associamo ad ogni pixel un bit in base al colore predominante nel quadratino	Togliendo il colore, rimane la codifica che letta come sequenza di bit è 111101110101	Facendo il procedimento inverso, sostituiamo ai bit il colore (sempre con la stessa regola: 0 → bianco, 1→ nero) e otteniamo
1 1 1 1 0 1 1 1 0 1 0 1	1 1 1 1 0 1 1 1 0 1 0 1	che non coincide esattamente con l'immagine iniziale.

Proviamo a ridurre la grandezza di un pixel ovvero ad aumentare il numero di quadratini della griglia da sovrapporre all'immagine.

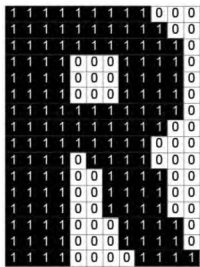
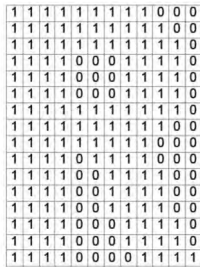



Immagine del carattere 'R'	Immagine con sovrapposta una griglia di 8x6 pixel
R	R

Ripetiamo gli stessi passi ...	Otteniamo la sequenza 111110 111110 110011 111110 111100 110110 110110 110011	che nuovamente trasformata diventa
1 1 1 1 1 0 1 1 1 1 1 0 1 1 0 0 1 1 1 1 1 1 1 0 1 1 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1	1 1 1 1 1 0 1 1 1 1 1 0 1 1 0 0 1 1 1 1 1 1 1 0 1 1 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1	

più fedele all'originale anche se con una “particolarità”. Domanda: sai spiegarne il perché?

Un ultimo raffinamento.

Immagine del carattere 'R'	Immagine con sovrapposta una griglia di 12x16 pixel
R	R

Ripetiamo gli stessi passi ...	Quale sequenza di bit otteniamo?	E infine ...
1 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 0 1 1 1 1 0 1 1 1 1 0 0 0 1 1 1 1 0 1 1 1 1 0 0 0 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 0 1 1 1 1 0 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0 0 0 1 1 1 1 0 1 1 1 1 0 0 0 1 1 1 1 0 1 1 1 1 0 0 0 0 1 1 1 1	1 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 0 1 1 1 1 0 1 1 1 1 0 0 0 1 1 1 1 0 1 1 1 1 0 0 0 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 0 1 1 1 1 0 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0 0 0 1 1 1 1 0 1 1 1 1 0 0 0 1 1 1 1 0 1 1 1 1 0 0 0 0 1 1 1 1	R

Non coincide certamente con l'immagine iniziale ma è "più simile" ad essa rispetto alle precedenti.
La codifica dei soli colori nero e bianco è accettabile se l'immagine è un carattere, non lo è se l'immagine è una fotografia, un disegno, un dipinto. Infatti anche una fotografia in bianco e nero di fatto non contiene solo questi 2 colori ma diverse tonalità di grigio (per le sfumature, i contrasti, le ombre e quant'altro). Immaginiamo allora di utilizzare 4 bit per codificare il colore di un singolo pixel: significa avere a disposizione 16 ($=2^4$) diverse combinazioni di 0 e 1, ognuna delle quali può essere associata ad un pixel e, quindi, ogni pixel può assumere un colore che varia tra i 16 possibili. Più precisamente non si parla di 16 colori ma di 16 tonalità di grigio che vanno da 0000 (bianco) a 1111 (nero) passando per varie sfumature di grigio (alla codifica 0001 corrisponderà un grigio chiarissimo, alla codifica 1110 un grigio molto scuro).

A questo punto è chiaro che, aumentando il numero di bit impiegati per la codifica del colore, aumenta la quantità di toni di grigio che possono essere rappresentati: con 8 bit è possibile rappresentare 256 ($=2^8$) toni di grigio.
Ma, se vogliamo codificare un'immagine con i suoi colori, occorre approfondire alcuni concetti riguardanti la teoria del colore.

La rappresentazione RGB del colore

La "teoria del colore" afferma che una parte piuttosto ampia dei colori percepiti dai nostri occhi può essere immaginata come sovrapposizione di TRE componenti monocromatiche dette PRIMARIE; il lettore avrà certamente avuto modo di sperimentare come, nella pittura, alcuni particolari tipi di ROSSO, BLU e GIALLO si prestino ad essere combinati tra loro per riprodurre gli altri colori: il BIANCO viene usato come una sorta di diluitore cioè di dispersore del pigmento colorato con l'obiettivo di ottenere tonalità più chiare, mentre il NERO è utilizzato per scurire i toni; la necessità del "nero" (a dispetto della teoria) scaturisce dal fatto che il miscuglio in parti uguali dei tre colori primari produce il cosiddetto BISTRO, un nero piuttosto imperfetto a causa delle impurità presenti nei colori primari. Anche nella tecnica tipografica si fa uso dello stesso principio adottando il MAGENTA (un rosso violaceo), il CIANO (un celeste) e il GIALLO come colori primari con l'aggiunta del NERO necessario per aggiungere "forza" alle tonalità scure ottenute con la sovrapposizione dei tre colori primari; nella tecnica tipografica si parla di "quadricromia" e si usa l'acronimo "CMYK" intendendo proprio "C" per "cyan" (ciano), "M" per "magenta", "Y" per "yellow" (giallo) e infine "K" che, nonostante in origine indicasse altro ("Key plate", tecnica di allineamento delle lastre CMY), nella prassi corrente si intende per "nero".

Tanto in pittura che in tecnica tipografica, tuttavia, il colore viene riprodotto per SINTESI SOTTRATTIVA in quanto ciò che i nostri occhi vedono e percepiscono come colore è la parte riflessa della luce, incidente su ciò che osserviamo: la parte non riflessa viene in effetti assorbita, dunque SOTTRATTA, dal pigmento colorato.
In effetti siamo in grado di "vedere" un quadro oppure un'immagine stampata SOLO se siamo in un ambiente luminoso: riuscireste a vedere qualcosa in una stanza perfettamente al buio?

Ben diversa è la situazione quando guardiamo lo schermo del computer oppure un televisore: a differenza di quadri e stampe lo schermo EMETTE la luce. Si parla allora di riproduzione del colore per SINTESI ADDITIVA in quanto ciò che vediamo in effetti è la sovrapposizione di tre sorgenti luminose primarie che in questo caso sono state individuate nella luce ROSSA, nella luce VERDE e nella luce BLU unite nella famosa sigla RGB ("Red", "Green", "Blue").

Nella codifica RGB per ogni pixel occorre distinguere tre distinti valori riferibili, nell'ordine, alle componenti di luci rossa, verde e blu; utilizzando un byte per ogni colore primario ogni pixel è codificato con 3 byte (3*8=24 bit in tutto). Ogni gruppo

di 8 bit è relativo ad un colore primario e può essere pensato come la quantità di quel colore nell'insieme; i primi 8 bit rappresentano quindi la quantità di rosso che c'è in quel pixel, i successivi 8 la quantità di verde e gli ultimi 8 la quantità di blu. Oltre che con la codifica binaria, ogni pixel può essere rappresentato con la corrispondente codifica decimale o esadecimale (in entrambe le codifiche vengono utilizzati 3 numeri, uno per ciascuna delle 3 sequenze di 8 bit), migliorando così la leggibilità della rappresentazione.

Facciamo qualche esempio:

COLORE	BINARIO			DECIMALE			ESADECIMALE		
	R	G	B	R	G	B	R	G	B
rosso	1 1 1 1 1 1 1 1	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	255	0	0	FF	00	00
verde	0 0 0 0 0 0 0 0	1 1 1 1 1 1 1 1	0 0 0 0 0 0 0 0	0	255	0	00	FF	00
blu	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	1 1 1 1 1 1 1 1	0	0	255	00	00	FF

Osserviamo che il giallo, colore primario percepito dall'occhio umano, non è presente nella codifica RGB, ma è rappresentabile come combinazione di rosso e verde; facciamo qualche esempio, iniziando con la codifica di una tonalità più luminosa di giallo, passando per una tonalità più scura e finire con una tonalità più chiara:

COLORE (giallo)	BINARIO			DECIMALE			ESADECIMALE		
	R	G	B	R	G	B	R	G	B
luminoso	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1	0 0 0 0 0 0 0 0	255	255	0	FF	FF	00
scuro	1 1 1 0 0 1 1 0	1 1 1 0 0 1 1 0	0 1 0 0 1 1 0 0	230	230	102	E6	E6	4C
chiaro	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1	0 1 1 0 0 1 1 0	255	255	102	FF	FF	66

Utilizzando 3 byte, quindi 24 bit, possiamo codificare ben 2^{24} = 16.777.216 colori distinti:
un'enormità, tanto da ipotizzare, nel criterio di codifica, l'uso di un numero minore di bit senza un sensibile degrado della percezione di un'immagine ma con il vantaggio di un minore “peso” in termini di quantità di byte.

Un altro modo per contenere il numero di byte necessari per memorizzare un'immagine a colori è quello di utilizzare un sottoinsieme dei 16.777.216 colori distinti. I colori selezionati vengono memorizzati in una “tavolozza” di colori o “palette”, ogni colore è codificato con lo standard RGB e occupa una posizione nella tavolozza; ad ogni pixel è associato un numero che non rappresenta più il colore del pixel nell'immagine ma la posizione che ha il colore del pixel in esame nella tavolozza del colore.
Ovviamente oltre alla stessa immagine memorizzata bisognerà anche salvare la palette, una sorta di codice dei colori utilizzati che consentirà la giusta visualizzazione.

Calcoliamo quanta memoria occorre per l'immagine : 12 (pixel) x 3 bit = 36 bit

per la palette: 8 x 24 bit =192 bit

dove: 8 = numero di elementi E 24= numero di bit per la codifica in RGB

totale: **228 bit (= 36+192)**

In RGB abbiamo la seguente codifica

pixel 1	pixel 2	pixel 3	pixel 4	pixel 5	pixel 6	pixel 7	pixel 8	pixel 9	pixel 10	pixel 11	pixel 12
FF0000	13F345	F6BCE6	9068E5	FFFF33	CDE6F6	AB1080	FFAA55	FF0000	FFFF33	CDE6F6	9068E5

Calcoliamo quanta memoria occorre per l'immagine (solo per l'immagine)

12 (pixel) x 24 bit(codifica RGB)= 288 bit

Generalmente, con la tavolozza c'è un risparmio nell'occupazione di memoria.

Domanda: ma se il colore che vogliamo rappresentare non è presente nella tavolozza?
Possiamo scegliere tra due possibili soluzioni: selezioniamo la tonalità, presente nella tavolozza, più vicina a quella da rappresentare accettando un margine di errore (approssimazione) oppure cambiamo palette.

Grafica vettoriale (cenni)

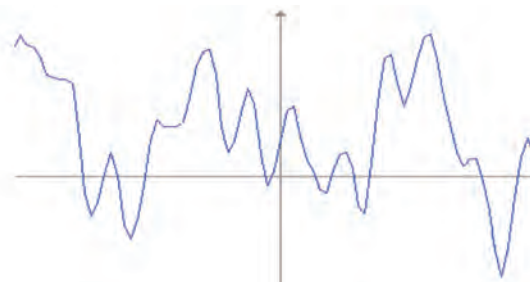
La codifica delle immagini mediante la grafica bitmap è adatta ormai solo ad alcune tipologie di immagini, sostituita nelle memorizzazioni di fotografie da formati compressi come per esempio il JPG, inoltre non si adatta a memorizzare disegni tecnici, diagrammi, mappe; in questi casi si preferisce utilizzare la grafica vettoriale che si basa sulla considerazione che ogni immagine può essere vista come un insieme di oggetti, i più utilizzati sono i luoghi geometrici come i punti e le rette.
Quindi, dopo aver trasformato l'immagine che si intende digitalizzare in un ben determinato insieme di oggetti, si memorizzano le coordinate di tali elementi; occorrerà poi un programma in grado di leggere tali dati e trasformarli in pixel perché l'immagine iniziale venga riprodotta.

L'utilizzo della grafica vettoriale, oltre a comportare una minore occupazione di memoria rispetto alla memorizzazione della stessa immagine con la grafica bitmap, permette una maggiore manipolazione poiché è possibile modificare la dimensione dell'immagine (ingrandendo o rimpicciolendo) senza perdita di nitidezza; non è un caso che le mappe che è possibile consultare sul Web o sui navigatori GPS sono immagini vettoriali.

D'altra parte la codifica delle immagini con la grafica vettoriale dipende dal software utilizzato, di conseguenza non esiste uno standard universalmente riconosciuto.

Un'ultima nota: anche nel caso di grafica vettoriale c'è il problema di riprodurre il colore, ma la trattazione dell'argomento esula dagli obiettivi di questi appunti.

Codifica dell'audio

Da un punto di vista fisico un suono può essere definito come una sensazione percepita dall'orecchio a causa del movimento di un corpo che mette in oscillazione l'aria. Tale movimento, detto più correttamente vibrazione, ha un andamento ondulatorio nel tempo e perciò un suono può essere rappresentato da un'onda, onda sonora.

Un suono è caratterizzato dalla *frequenza*, misurata in Hertz (Hz), che definisce l'altezza del suono, *dall'ampiezza* dell'onda, misurata in decibel (dB), che determina l'intensità del suono ed infine dalla *durata*.
La memorizzazione dei suoni su un disco di vinile può essere considerata una rappresentazione analogica del suono perché fornisce una rappresentazione continua dell'onda sonora, ma, ovviamente, non può essere memorizzata su un file ed elaborata da un computer.
E' necessario quindi convertire il segnale sonoro, analogico, in un segnale digitale; una tecnica utilizzata per la conversione analogico-digitale di un suono è quella di trasformare l'onda (il continuo) in un serie di valori rilevati in istanti di tempo distinti, di mantenere tali valori all'interno di un intervallo e codificare ciascun valore con un numero.
Ti ricorda qualcosa? Ebbene sì, è esattamente quanto già analizzato in dettaglio per la codifica delle immagini con la grafica bitmap; il processo di cui stiamo parlando è quello che, partendo dal campionamento del segnale e passando per la quantizzazione, arriva alla codifica dello stesso.

Non ripetiamo quanto già detto: con le dovute modifiche il procedimento è stato già descritto. Vogliamo solo ricordare alcuni dettagli:
- la griglia che si sovrappone al segnale digitale non ha carattere spaziale ma temporale;
- anche in questo caso il processo di quantizzazione introduce un errore come sempre avviene quando si passa da una grandezza continua ad un insieme discreto di numeri;
- maggiore è il numero di campioni, migliore è la qualità del suono riprodotto ma, ovviamente, maggiore è la quantità di memoria necessaria.

In ultimo, facciamo un po' di conti: nel processo di trasformazione il segnale originale viene trasformato in una sequenza di misure e, dunque, di numeri rappresentabili ovviamente in codifica binaria: 16 bit per ogni singolo campione (quindi 65.536 livelli distinti di ampiezza) sono solitamente sufficienti per una buona qualità di codifica del segnale acustico insieme ad una frequenza di campionamento di 44.100 Hz (cioè 44.100 campioni ogni secondo) utilizzata per una qualità CD (compact-disk).

Passiamo ai calcoli: quanti byte sono indispensabili per codificare quattro minuti di segnale audio (per esempio un brano musicale)? Presto detto:

Tempo (secondi)	N° totale di campioni (=campioni x tempo)	Occupazione (in byte e in Megabyte) (=n° totale campioni x byte occupati da un campione)
240 (=4 minuti * 60)	10.584.000 (=44.100*240)	21.168.000 (=10584000*2) = 20,18 MB (circa)

Quindi, un brano musicale di 4 minuti necessita di oltre 20 megabyte di codifica per una qualità audio di tipo CD (compact-disk).

Codifica video

Alla base della codifica dei "filmati" c'è la riflessione, antica quanto la storia del cinema, che, osservando in rapida sequenza immagini fotografiche (dette "frame") contenenti piccole variazioni, si ha la percezione della continuità e quindi del movimento. Una buona velocità in tal senso è 30 frame al secondo.
Se poi aggiungiamo una colonna sonora sincronizzata con le immagini otteniamo un video "completo" (suono+ immagine). Allo stesso modo, sequenze di immagini digitalizzate producono la stessa percezione: possiamo dunque pensare un video digitale come un oggetto codificato in un file contenente, in successione, i vari fotogrammi che compongono la sequenza video, a cui aggiungere i dati relativi alla digitalizzazione dei suoni (siano essi musica, voce umana,etc.) per completare il prodotto.

La digitalizzazione di un video è quindi una combinazione di tecniche di digitalizzazione di suoni e immagini; tra le inevitabili complicazioni insite in questo processo, vogliamo soffermare la nostra attenzione sul seguente quesito: quanto spazio occupa in termini di byte un filmato?
Facciamo alcune ipotesi e... qualche conto!
Immaginiamo che un filmato (senza sonoro per semplicità di calcolo) di 10 minuti sia composto da sequenze di immagini in alta definizione 1280x720 (per un totale di 921.600 pixel).
Ipotizziamo una codifica del colore RGB su 24 bit (3 byte): ogni frame è codificato su 3x921.600 byte cioè 2.764.800 byte.

Se consideriamo 30 frame al secondo, equivalendo 10 minuti a 600 secondi, otteniamo un numero di frame complessivo pari a 18.000 e quindi 49.766.400.000 byte che valgono più di 46 GB (gigabyte): una quantità di memoria enorme per soli 10 minuti di filmato.

In generale, per calcolare l'occupazione di memoria di un file che contiene la digitalizzazione di un video occorre determinare quanto spazio occupa ogni unità elementare in cui è stato scomposto il video (immagini, suoni, caratteri), contare quante unità elementari di ciascun tipo sono presenti, calcolare l'occupazione totale di ciascun tipo e infine sommare i valori ottenuti.

Classificazione dell'informazione

Giunti a questo punto, torniamo sulla prima definizione del capitolo che qui riportiamo arricchendola con l'aggiunta di un solo aggettivo:
"L'informatica si occupa della codifica BINARIA, della memorizzazione, della trasmissione e dell'elaborazione dell'informazione".
Il riferimento all'uso generalizzato della rappresentazione binaria dell'informazione dovrebbe ora risultare chiaro: ogni genere di informazione trattata da sistemi automatici DEVE essere rappresentabile come sequenza di BIT.

Viceversa per poter riprodurre l'informazione codificata in una sequenza di bit è necessario conoscere i criteri adottati nella codifica. Una sequenza di bit del tipo:

1010111101110110100011010111101110110110011010011001010100100001010011101 00
1010101110100000011010101101111111100110110000011010111111111011110011111 1001
1000101010101010000001010101000101011110110101110010010111010100011101010101

non è interpretabile fintanto che non viene assegnato un criterio per decifrarla, ossia un criterio di decodifica. Ricordando la distinzione tra dato e informazione possiamo dire che

informazione = dato (ovvero sequenza di bit) + criterio di decodifica

Solo quando è stabilito il criterio di codifica (e dunque di decodifica) ha dunque senso parlare di digitalizzazione dell'informazione secondo il seguente schema:

INFORMAZIONE > CODIFICA > INFORMAZIONE CODIFICATA
INFORMAZIONE CODIFICATA > DECODIFICA > INFORMAZIONE.

Un'ultima nota: l'informazione codificata viene memorizzata in un "oggetto" chiamato file a cui si assegna un NOME allo scopo di caratterizzare in modo chiaro ciò che rappresenta; nel caso per esempio di un documento di testo contenente il curriculum vitae di un tale "Mario", codificato in ASCII, si può assegnare un nome del tipo:

curriculum_mario.txt

dove si distinguono due parti, la prima "curriculum_mario" detta propriamente NOME; la seconda "txt" si dice ESTENSIONE e segue il PUNTO che fa da SEPARATORE.

In ambiente WINDOWS l'estensione indica il TIPO di informazione (quindi il criterio utilizzato nella codifica) che con una convenzione piuttosto diffusa nel mondo dell'Informatica indica un contenuto di tipo testuale "semplice" cioè codificato con l'uso del codice ASCII. Il nome "curriculum_mario", più che avere valore per l'operazione di decodifica, ha valore di "descrittore del contenuto" ed è liberamente scelto dall'autore dell'oggetto come "promemoria":
in questo modo, chi si trovasse a leggere nell'elenco dei file:

curriculum_mario.txt
curriculum_cesare.txt
curriculum_rosa.txt

dedurrebbe senza difficoltà il tipo e il contenuto dei file.
Nella seguente tabella si indicano le estensioni più utilizzate insieme al tipo di informazione che generalmente caratterizzano:

Tipologia	Estensioni
Grafica bitmap o raster	bmp, jpg, gif, tiff, png, ico
Grafica vettoriale	eps, ai, wmf
File eseguibili, codice oggetto, librerie dinamiche	exe, com, lib, jar, war, ear, class, dll
Documenti	doc, docx, html, rtf, odt, txt
Audio e musica	wav, wma, au, mp2, mp3, mid
Archiviazione e compressione	zip, 7z, jar, arc, gzip, tar, tar.gz, rar
Linguaggi di programmazione	c, cpp, for, java, lisp, cob, asm, bas, php, asp, jsp, pl, py, rb
Linguaggi di descrizione e stilizzazione pagina	pdf, postscript, xsl-fo, css, xslt/xsl

Approfondimento: aritmetica binaria

Il sistema di numerazione binario è, come sottolineato più volte, un sistema di numerazione posizionale: di conseguenza in esso le quattro operazioni fondamentali seguono le stesse regole applicate nel sistema decimale.
(Per semplicità, supponiamo che non ci siano limiti circa il numero di bit che occorrono per rappresentare il risultato).

Addizione: si sommano le cifre di uguale peso, ricordando che si può avere un riporto (*carry*) sul bit di peso immediatamente superiore.

+	0	1	
0	0	1	
1	1	0	← con riporto di 1

Sottrazione: si sottraggono le cifre di uguale peso, ricordando che si può avere un prestito (*borrow*) sul bit di peso immediatamente superiore
(*) sulla prima colonna a sinistra le cifre del minuendo, sulla prima riga le cifre del sottraendo

- (*)	0	1	
0	0	1	← con prestito di 1
1	1	0	

Moltiplicazione: le cifre costituenti il primo numero (moltiplicando) vengono moltiplicate per ciascuna cifra del secondo numero (moltiplicatore): se la cifra del moltiplicatore è pari a 1 si riscrive il moltiplicando, in caso contrario (cifra=0) il risultato è pari a 0. Si sommano tutti i numeri ottenuti (attenzione ai riporti!)

*	0	1
0	0	0
1	0	1

Divisione: anche in questo caso il procedimento coincide con quello applicato tra numeri in base 10, con la semplificazione che il gruppo di cifre del dividendo esaminato può contenere il divisore 1 o 0 volte.

Esempi:

Addizione: $110100_2 + 111000_2$

```
 1 1
   1 1 0 1 0 0 +
   1 1 1 0 0 0 =
 1 1 0 0 1 0 0
```

$110100_2 + 111000_2 = 1100100_2$

Sottrazione: $10011_2 - 1111_2$

```
      1
  0  10  10
  1   0   0  1  1 -
      1   1  1  1 =
  0   0   1  0  0
```

$10011_2 - 1111_2 = 100_2$

Moltiplicazione: $11101_2 * 1110_2$

```
                 1 1 1 0 1 *
                   1 1 1 0 =
 1  10  10  1  1
                 0 0 0 0 0
               1 1 1 0 1
             1 1 1 0 1
           1 1 1 0 1
         1 1 0 0 1 0 1 1 0
```

$11101_2 * 1110_2 = 110010110_2$

Divisione: $110101_2 : 11_2$

```
1 1 0 1 0 1 | 1 1
1 1         | 1 0 0 0 1
0 0 0
  0 0
  0 0 1
    0 0
    0 1 0
      0 0
      1 0 1
        1 1
      0 1 0
```

$110101_2 : 11_2 = 10001_2$ (resto = 10_2)

Interi positivi e negativi

Torniamo sulla sottrazione; non è un caso che tutti gli esempi proposti sono caratterizzati da un valore del minuendo maggiore del valore del sottraendo, poiché finora abbiamo considerato la codifica e, quindi le operazioni, tra numeri naturali (numeri interi senza segno). Il passo successivo è quello esaminare la codifica dei numeri con segno, cioè dei numeri interi (positivi, negativi e 0).
Nota bene: le argomentazioni successive hanno valore solo se si prendono in esame valori numerici codificati con un numero prefissato di cifre.
Tra i diversi metodi per rappresentare i numeri interi in base 2, prendiamo in considerazione
• la rappresentazione in modulo e segno (M&S)
• la rappresentazione in complemento a 2

Rappresentazione in modulo e segno

Dopo aver stabilito il numero N di bit che si vogliono utilizzare per la codifica, si riserva il bit più significativo per il segno, tutti gli altri (N-1) sono utilizzati per codificare il valore assoluto del numero in esame.
La convenzione adottata è di utilizzare il valore 1 per il segno negativo, 0 per quello positivo.

Esempio: con N=4 bit
numero = -3 → **1**011 dove **1** rappresenta il segno -
numero = +3 → **0**011 dove **0** rappresenta il segno +
→ i rimanenti 3 bit (011) rappresentano il valore assoluto, ovviamente uguale in entrambe le codifiche

Osservazioni:
- lo 0 ha una doppia rappresentazione, una positiva e una negativa;
- l'intervallo di valori rappresentabili con N bit è $[-2^{N-1}+1, +2^{N-1}-1]$.

Alla facilità di codifica corrispondono però problemi di elaborazione: il fatto che il bit più significativo (il bit del segno) ha un significato diverso dagli altri implica che deve essere trattato in modo diverso nelle operazioni.

Esempio: Calcoliamo -20 +2 (supponiamo di utilizzare codifiche su 8 bit)
-20 = 10010100 quindi 10010100 +
+ 2 = 00000010 00000010 =
10010110

che corrisponde a -22 (non a -18 che è il risultato corretto)

Il problema deriva dal fatto che per il bit del segno non vale più la regola del peso associato alla posizione.
Si preferisce utilizzare, invece, la notazione in complemento a 2, che ci permette di trasformare la sottrazione in addizione.

Rappresentazione in complemento a 2

Definiamo il significato di complemento ad una base:
in generale si definisce complemento ad una base M di un numero X rappresentato con N cifre, quel numero Y che sommato a X dà come risultato la potenza N-esima della base.
Difficile? No, se rileggiamo quanto detto attraverso degli esempi:

esempio 1:	se X= 6, M= 10 e N= 1, la potenza 10^1 è 10; quanto vale Y, ovvero quale è quel numero che sommato a 6 dà come risultato 10?
	la risposta, ovvia, è Y= 4
esempio 2:	se X= 38, M= 10 e N= 2, la potenza 10^2 è 100; quanto vale Y?
	la risposta è Y= 62 (62+38 = 100).

E se la base è 2? Il procedimento è il medesimo; esaminiamolo attraverso un esempio:
se X= 11001 e N= 5, la potenza 2^5 è 100000, quindi il numero che sommato a X dà come risultato 100000 è Y=00111.
Infatti 11001+
00111 =
100000

Leggiamo quanto svolto in base 10: 11001_2= 25, la potenza 2^5 è 32 =100000 (ricordati che devi ragionare in base 2).
Quale è quel numero che sommato a 25 dà 32?
Risposta: 7 che in base 2 è proprio 111.
Introduciamo ora la **rappresentazione** dei numeri interi in complemento a 2 (*con N bit*).

Definizione:
• la rappresentazione di un numero positivo coincide con la rappresentazione in modulo e segno (entrambe con N bit)
• la rappresentazione di un numero negativo è costituita dal complemento a 2 del corrispondente numero positivo.

Osservazioni:
• l'intervallo di valori rappresentabili con N bit è $[-2^{N-1}, +2^{N-1}-1]$, quindi è possibile rappresentare un valore in più rispetto alla rappresentazione in modulo e segno.
• i numeri positivi iniziano tutti con un bit a 0
• i numeri negativi iniziano tutti con un bit a 1
• esiste una sola rappresentazione dello 0: (0 ... 0).

Riportiamo nella successiva tabella la codifica in modulo e segno e in complemento a 2 dei numeri interi rappresentabile con N=4 bit.

in base 10	modulo e segno	complemento a 2
-8	———	1000
-7	1111	1001
-6	1110	1010
-5	1101	1011
-4	1100	1100
-3	1011	1101
-2	1010	1110
-1	1001	1111
-0	1000	———
+0	0000	0000
+1	0001	0001
+2	0010	0010
+3	0011	0011
+4	0100	0100
+5	0101	0101
+6	0110	0110
+7	0111	0111

Esiste una regola di facile applicazione che permette di scrivere la rappresentazione di un numero negativo in complemento a 2 con N bit:
1. si scrive la codifica con N bit del numero positivo corrispondente
2. si lascia inalterata, partendo dal bit meno significativo, la sequenza di bit fino al primo 1 che si incontra, che rimane anch'esso invariato; si modificano tutti i altri bit: 0 diventa 1, 1 diventa 0.

Esempio: vogliamo codificare in base 2 con 6 bit il corrispondente valore di -22 in base 10
1° passo: $+ 22_{10} = 010110_2$
2° passo: si complementa bit a bit, applicando la regola appena esposta

0 1 0 1 1 0
↓ ↓ ↓ ↓ ↓ ↓
1 0 1 0 1 0

quindi $-22_{10}=101010_2$ in complemento a 2

Affrontiamo ora il problema posto all'inizio del paragrafo:
la sottrazione tra numeri relativi.

Partiamo dalla considerazione che la differenza tra due numeri equivale alla somma algebrica tra il primo operando e l'opposto del secondo.

Infatti 23 -
12 = → equivale 23 + (-12) = +11
+11

È sufficiente rappresentare il sottraendo in complemento a 2, trasformando così una sottrazione in un'addizione.
Enunciamo la regola per eseguire la somma algebrica tra 2 numeri interi codificati in complemento a 2 su N bit:

- si esprimono i numeri in complemento a 2 su N bit (la codifica differisce solo per inumeri negativi)
- si esegue la somma
- si trascura l'eventuale overflow (ovvero il bit di posizione N+1)

Applichiamo la regola (negli esempi successivi i numeri sono codificati su 6 bit)

Esempio: Operandi di segno opposto in questo caso il valore ottenuto è sempre corretto (il numero di bit è sufficiente per memorizzare il risultato)

-22 = 101010 101010 +
+ 2= 000010 000010 =
101100

Nel risultato il bit del segno è 1 → valore negativo, quindi occorre complementare 101100 diventa 010100 = 20
risultato finale [-22 + 2 = -20] → risultato corretto

Esempio: Operandi di segno concorde + **è sempre senza overflow** (perché?)
in questo caso il valore ottenuto non sempre è corretto

risultato corretto (senza overflow)		risultato errato (senza overflow)	
+17 + 8 = +25		+17 +16 = +33	
+17 = 010001	010001 +	+17 = 010001	010001 +
+ 8 = 001000	010000 =	+16 = 010000	001000 =
	011001		100001

poiché [$011001_2 = 25_{10}$] Nel risultato il bit del segno è 1 → valore negativo
risultato corretto risultato errato

Esempio: Operandi di segno concorde - **è sempre con overflow** (perché?)
in questo caso il valore ottenuto non sempre è corretto

risultato corretto (con overflow)		risultato errato (con overflow)	
-24 + (-7) = -31		-24 + (-12) = -36	
-24 = 101000	101000 +	-24 = 101000	101000 +
-7 = 111001	110100 =	-12 = 110100	110100 =
	1100001		1011100

Il bit più significativo è il bit di overflow → si trascura (è il 7° bit), il bit immediatamente successivo è il bit del segno: valore 1 quindi il risultato si complementa

Il bit più significativo è il bit di overflow → si trascura (è il 7° bit), il bit immediatamente successivo è il bit del segno: valore 0 quindi il risultato è positivo

poiché [$1100001_2 = -31_{10}$]
risultato corretto risultato errato

Esercizi: eseguire le seguenti operazioni in aritmetica binaria, codificare i numeri in decimale e controllare i risultati ottenuti

addizione

1) $111_2 + 1_2$ [8_{10}]
2) $1111_2 + 1101_2$ [28_{10}]
3) $1001001_2 + 1110_2$ [87_{10}]
4) $1111101_2 + 11111111_2$ [380_{10}]
5) $1000000_2 + 10000000_2$ [192_{10}]
6) $1100100_2 + 11011_2$ [127_{10}]
7) $1000000_2 + 11111_2$ [95_{10}]

sottrazione(tra numeri naturali)

8) $100000_2 - 10000_2$ [16_{10}]
9) $100110_2 - 110_2$ [32_{10}]
10) $1111000_2 - 10100_2$ [100_{10}]
11) $1001000_2 - 1000001_2$ [7_{10}]
12) $1000110_2 - 110010_2$ [20_{10}]
13) $1011011_2 - 100110_2$ [53_{10}]
14) $1011000_2 - 11010_2$ [62_{10}]

moltiplicazione

15) $101_2 * 10_2$ [10_{10}]
16) $101_2 * 110_2$ [30_{10}]
17) $1010_2 * 111_2$ [70_{10}]
18) $11011_2 * 1100_2$ [324_{10}]
19) $11111_2 * 111_2$ [217_{10}]
20) $11111_2 * 1111_2$ [465_{10}]
21) $11111_2 * 11111_2$ [961_{10}]

divisione

22) $1100_2 : 11_2$ [Q= 4_{10} ; R= 0_{10}]
23) $101111_2 : 101_2$ [Q= 9_{10} ; R= 2_{10}]
24) $101111_2 : 111_2$ [Q= 6_{10}; R= 5_{10}]
25) $1010000_2 : 1011_2$ [Q=7_{10} ; R= 3_{10}]
26) $1111101_2 : 110_2$ [Q= 20_{10}; R= 5_{10}]
27) $10001000_2 : 1111_2$ [Q= 9_{10}; R= 1_{10}]
28) $100000000_2 : 10000_2$ [Q= 16_{10}; R= 0_{10}]

somma algebrica (in complemento a 2 su 7 bit)

29) $0100011_2 + 0101101_2$ [-48_{10} errore]
30) $0100011_2 + 0001100_2$ [47_{10}]
31) $1010011_2 + 0010111_2$ [-22_{10}]
32) $0110011_2 + 1000100_2$ [-9_{10}]
33) $0011011_2 + 1100100_2$ [-1_{10}]
34) $1000100_2 + 1001110_2$ [-46_{10} errore]
35) $1110001_2 + 1001111_2$ [-64_{10}]
36) $1101110_2 + 1110010_2$ [-32_{10}]

Laboratorio: Conversione con OpenOffice Calc

Conversione dei numeri da un sistema numerico ad un altro

Ora proviamo a costruire un foglio di calcolo che mi permetta di convertire i numeri nei vari sistemi numerici. Come ipotesi di partenza per tutti gli esercizi che realizzeremo, consideriamo sempre un numero con massimo 6 cifre, ma tale foglio di calcolo potrà essere ampliato a vostro piacimento utilizzando un numero diverso di cifre.

Conversione Binario → Decimale

Innanzitutto apriamo il foglio di calcolo e scriviamo le seguenti informazioni:
Con lo schema impostato come figura riusciremo a convertire il numero binario "011101" in un numero decimale.

	A	B	C	D	E	F	G	H
1	CONVERSIONE DI UN NUMERO							
2	da:							a:
3	BINARIO							DECIMALE
4	0	1	1	1	0	1		
5	5	4	3	2	1	0	POSIZIONE	
6	2	2	2	2	2	2	BASE	
7							POTENZA	
8							PRODOTTO	

	A	B	C	D	E	F	G	H
1	CONVERSIONE DI UN NUMERO							
2	da:							a:
3	BINARIO							DECIMALE
4	0	1	1	1	0	1		29
5	5	4	3	2	1	0	POSIZIONE	
6	2	2	2	2	2	2	BASE	
7	32	16	8	4	2	1	POTENZA	
8	0	16	8	4	0	1	PRODOTTO	

Per prima cosa inserire tutte le informazioni cosi come scritte in figura.
Nella cella A7 scrivere la formula =A6^A5 e trascinare fino alla cella F7
Nella cella A8 scrivere la formula =A7*A4 e trascinare fino alla cella F8
Nella cella H4 scrivere la formula = A8+B8+C8+D8+E8+F8 e premere INVIO

Ora sappiamo che "011101" corrisponde a "29" nel sistema decimale!

Conversione Ottimale → Decimale

Proviamo ora ad impostare un foglio di calcolo per effettuare la conversione da un numero ottale in decimale:
Per esempio, vogliamo trovare il corrispondente numero decimale del numero "256" in base 8.
Partiamo sempre costruendo una tabella come la figura seguente:

	A	B	C	D	E	F	G	H
1	CONVERSIONE DI UN NUMERO							
2	da:							a:
3	OTTALE							DECIMALE
4				2	5	6		
5	5	4	3	2	1	0	POSIZIONE	
6	8	8	8	8	8	8	BASE	
7							POTENZA	
8							PRODOTTO	

Successivamente inserite le seguenti informazioni:
Nella cella A7 scrivere la formula =A6^A5 e trascinare fino alla cella F7
Nella cella A8 scrivere la formula =A7*A4 e trascinare fino alla cella F8
Nella cella H4 scrivere la formula = A8+B8+C8+D8+E8+F8 e premere INVIO.

Ora sappiamo che il numero “256” in ase 8 corrisponde al numero 174 nel sistema decimale.

A	B	C	D	E	F	G	H
CONVERSIONE DI UN NUMERO							
da:							a:
OTTALE							DECIMALE
			2	5	6		174
5	4	3	2	1	0	POSIZIONE	
8	8	8	8	8	8	BASE	
32768	4096	512	64	8	1	POTENZA	
0	0	0	128	40	6	PRODOTTO	

Conversione Esadecimale → Decimale

Per convertire un numero in sistema esadecimale in decimale, le cose si complicano un poco perchè le cifre del sistema esadecimale non sono tutte numeriche, ma dopo la cifra 9 dobbiamo considerare A, B, C, D, E, F che rispettivamente corrispondono a 10, 11, 12, 13, 14, 15. Procediamo quindi nel seguente modo:
Per prima cosa predisponiamo nel nostro foglio Calc nelle due colonne J e K l’associazione simbolo esadecimale ⟷ valore
Successivamente impostiamo la seguente tabella:

J	K
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
A	10
B	11
C	12
D	13
E	14
F	15

	A	B	C	D	E	F	G	H
1	CONVERSIONE DI UN NUMERO							
2	da:							a:
3	ESADECIMALE							DECIMALE
4				1	4	A		
5							NUMERO CORRISPONDENTE	
6	5	4	3	2	1	0	POSIZIONE	
7	16	16	16	16	16	16	BASE	
8							POTENZA	
9							PRODOTTO	

Nelle celle A5:F5 dobbiamo impostare una Funzione che mi permetta di trovare la cifra numerica corrispondente alla cifra esadecimale inserita. La funzione in questione è **=CERCA.VERT( )**. Come funziona? Questa funzione, in generale, permette di visualizzare un dato che si trova sullo stesso rigo di un valore dato in input. La sintassi della funzione è:
=CERCA.VERT(valore da cercare; tabella dei valori; colonna dove si trova il valore da visualizzare).

Quindi, il valore da cercare lo abbiamo inserito nelle caselle gialle; la tabella dei valori si trova nelle celle J1:K16 e l'informazione che ci serve si nella colonna 2 della stessa tabella J1:K16.

Nella cella A5, quindi, proviamo ad inserire la funzione:
=CERCA.VERT(A4;$J$1:$K$16;2) e trascinatela fino alla cella F5.
Nella cella A8 scrivere la formula =A7^A6 e trascinare fino alla cella F8
Nella cella A9 scrivere la formula =A8*A5 e trascinare fino alla cella F9
Nella cella H4 scrivere la formula = A8+B8+C8+D8+E8+F8 e premere INVIO.

	A	B	C	D	E	F	G	H
1	CONVERSIONE DI UN NUMERO							
2	da:							a:
3	ESADECIMALE							DECIMALE
4				1	4	A		330
5	0	0	0	1	4	10	NUMERO CORRISPONDENTE	
6	5	4	3	2	1	0	POSIZIONE	
7	16	16	16	16	16	16	BASE	
8	1048576	65536	4096	256	16	1	POTENZA	
9	0	0	0	256	64	10	PRODOTTO	

Ora sappiamo che il numero in base esadecimale “14A” corrisponde al numero decimale “330”.

Conversione Decimale → Binario

Ora proviamo a fare il contrario. Conoscendo il valore in decimale vogliamo calcolare il corrispondente numero in binario. Visto che abbiamo già utilizzato il numero 29 nel primo esempio, proviamo a prendere in considerazione questo numero per effettuare una controprova.

Impostare il foglio secondo la seguente figura:

	A	B	C	D	E
1	CONVERSIONE DI UN NUMERO				
2					
3	DECIMALE		BINARIO		
4	29				
5					
6	DIVIDENDO	DIVISORE	RISULTATO	PARTE INTERA	RESTO
7		2			
8					

Nella cella A7 scriviamo =A4
Nella cella C7 scriviamo =A7/B7
Nella cella D7 scriviamo =INT(C7)
Nella cella E7 scriviamo =RESTO(A7;B7)

se avete fatto tutto bene, otterrete il risultato come in figura:

	A	B	C	D	E
1	CONVERSIONE DI UN NUMERO				
2					
3	DECIMALE		BINARIO		
4	29				
5					
6	DIVIDENDO	DIVISORE	RISULTATO	PARTE INTERA	RESTO
7	29	2	14,5	14	1
8					
9					

Ora nella cella A8 scriviamo =D8
Nella cella C8 scriviamo =A8/B8
Nella cella D8 scriviamo =INT(C8)
Nella cella E8 scriviamo =RESTO(A8;B8)

Ripetiamo la stessa operazione fino a quando la cella che contiene la "Parte Intera" raggiungerà il valore "0", come la seguente figura:

Ora bisogna fare in modo che resti delle varie divisioni vengano letti a partire dall'ultimo.

	A	B	C	D	E
1	CONVERSIONE DI UN NUMERO				
2					
3	DECIMALE		BINARIO		
4	29				
5					
6	DIVIDENDO	DIVISORE	RISULTATO	PARTE INTERA	RESTO
7	29	2	14,5	14	1
8	14	2	7	7	0
9	7	2	3,5	3	1
10	3	2	1,5	1	1
11	1	2	0,5	0	1

Possiamo usare per esempio la funzione =CONCATENA() che permette di unire in un'unica cella informazioni che si trovano in più celle.
Proviamo a scrivere nella cella C4 la seguente funzione =CONCATENA(E11;E10;E9;E8;E7).
Otterremo la seguente figura:

	A	B	C	D	E
1	CONVERSIONE DI UN NUMERO				
2					
3	DECIMALE		BINARIO		
4	29		11101		
5					
6	DIVIDENDO	DIVISORE	RISULTATO	PARTE INTERA	RESTO
7	29	2	14,5	14	1
8	14	2	7	7	0
9	7	2	3,5	3	1
10	3	2	1,5	1	1
11	1	2	0,5	0	1

Abbiamo, in questo modo, verificato che effettivamente il numero decimale "29" corrisponde al numero binario "11101".

Conversione Decimale → Ottimale

Per quanto riguarda l'impostazione del foglio per eseguire questa conversione procediamo come nell'esercizio precedente, solo che questa volta nella colonna DIVISORE dobbiamo digitare "8".
Otterrete il seguente risultato:

	A	B	C	D	E
1	CONVERSIONE DI UN NUMERO				
2					
3	DECIMALE		BINARIO		
4	174		00256		
5					
6	DIVIDENDO	DIVISORE	RISULTATO	PARTE INTERA	RESTO
7	174	8	21,75	21	6
8	21	8	2,63	2	5
9	2	8	0,25	0	2
10	0	8	0	0	0
11	0	8	0	0	0

Conversione Decimale → Esadecimale

Anche in questo caso procedete sempre allo stesso modo, modificando semplicemente la colonna relativa a divisore, inserendo questa volta il valore "16".

Otterrete il seguente risultato:

	A	B	C	D	E
1	CONVERSIONE DI UN NUMERO				
2					
3	DECIMALE		BINARIO		
4	330				
5					
6	DIVIDENDO	DIVISORE	RISULTATO	PARTE INTERA	RESTO
7	330	16	20,63	20	10
8	20	16	1,25	1	4
9	1	16	0,06	0	1
10	0	16	0	0	0
11	0	16	0	0	0

Per poter assegnare la lettera per i numeri maggiori di 9, bisogna utilizzare sempre il comando =CERCA.VERT() impostando a parte una tabella simile a quella utilizzata nell'esercizio di conversione da esadecimale a decimale.

Nella cella C4 bisognerà utilizzare la funzione =CONCATENA(E7;G1:H16;2). Otterremo, in questo modo la seguente figura:

	A	B	C	D	E	F	G	H
1	CONVERSIONE DI UN NUMERO						0	0
2							1	1
3	DECIMALE		BINARIO				2	2
4	330						3	3
5							4	4
6	DIVIDENDO	DIVISORE	RISULTATO	PARTE INTERA	RESTO		5	5
7	330	16	20,63	20	10	A	6	6
8	20	16	1,25	1	4	4	7	7
9	1	16	0,06	0	1	1	8	8
10	0	16	0	0	0		9	9
11	0	16	0	0	0		10	A
12							11	B
13							12	C
14							13	D
15							14	E
16							15	F

3. I connettivi logici dell'Algebra di Boole

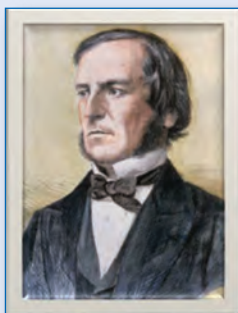
Autore: **Angelo Oliva**

Competenze	Abilità	Conoscenze
Interpretare situazioni complesse, composte da più condizioni, comprenderne il significato.	*Risolvere espressioni logiche a due e tre variabili, formulare proposizioni composte per selezionare particolari sott'insiemi di elementi su gruppi complessi*	*Elementi delle algebre di Boole e della logica delle proposizioni*

Introduzione

Nell'ambito dell'informatica in generale e della programmazione in particolare non si può prescindere dalla conoscenza dei principi della logica degli enunciati. Ad esempio, quando un'azione si può svolgere solo se si verificano due condizioni (se ho l'automobile e le catene a bordo potrò percorrere la strada innevata).

L'applicazione dei principi della logica degli enunciati non è prerogativa esclusiva della programmazione, ne è prova l'inserimento di tale argomento nel programma ufficiale per il conseguimento della patente europea per il computer, meglio nota con l'acronimo inglese ECDL e precisamente nel primo dei sette moduli del syllabus Ver. 5.0, Concetti di Base dell'ICT, paragrafo 1.0.3.2 "Distinguere il ruolo dei connettivi logici (AND, OR, NOT) nell'informatica."

In passato l'uso dei connettivi logici era necessario per svolgere delle ricerche avanzate su internet con i motori più conosciuti, per esempio per cercare informazioni su Ganimede inteso come pianeta e non come figura mitologica bisognava digitare nella barra d'inserimento della chiave di ricerca "*Ganimede AND satellite NOT mito*". Oggi, per semplificare le ricerche avanzate, i motori di ricerca prevedono la possibilità di perfezionare le ricerche attraverso un menù che facilita la costruzione della chiave di ricerca.

George Boole (1815 -1864)

Nel 1854 pubblicò la sua opera più importante, "An investigation into the Laws of Thought, on Which are founded the Mathematical Theories of Logic and Probabilities" indirizzata alle leggi del pensiero, con la quale propose una nuova impostazione della logica: scopo dell'opera fu di studiare le leggi delle operazioni mentali alla base del ragionamento esprimendole nel linguaggio simbolico del calcolo e di istituire, di conseguenza, una disciplina scientifica della logica sorretta da un metodo; dopo aver rilevato le analogie fra oggetti dell'algebra e oggetti della logica, ricondusse le composizioni degli enunciati a semplici operazioni algebriche. Con questo lavoro fondò la teoria di quelle che ora sono dette **algebre di Boole** (o semplicemente **algebra booleana**). Pur mantenendo distinte le operazioni mentali da quelle algebriche e le leggi logiche dai settori delle scienze naturali, il compito di Boole fu quello di travestire la logica con un abito matematico algebrico. I così detti "valori Booleani", sono utilizzati molto frequentemente nell'informatica e ne sono essenziali, dai linguaggi di basso livelli (**Assembly**), a quelli di alto livello e alle tecnologie web (**php**).

Le Proposizioni e i predicati

La logica delle proposizioni è un linguaggio formale regolato da una semplice struttura sintattica, basata su proposizioni elementari (atomi) che composte tramite connettivi logici restituiscono un valore di verità in base ai singoli valori di verità delle proposizioni connesse.

La Proposizione nell'ambito della logica delle proposizioni si distingue da quella del dizionario perché si definisce Proposizione (o Enunciato) una qualsiasi espressione autonoma e di senso compiuto alla quale si può attribuire un valore di verità Vero o Falso in modo oggettivo.

"Sta piovendo", "quella macchina è rossa" sono proposizioni valide perché si può affermare in modo incontrovertibile se sono vere o false, invece le espressioni "non fa molto freddo" o "il colore più bello è il rosso" pur potendo essere vere o false non sono oggettive ma dipendono dal parere del soggetto che le valuta.

Il Predicato è una proposizione contenente una o più variabili il cui valore di verità in un determinato istante dipende dai valori assunti dalle sue variabili nello stesso istante.

Partendo dalla proposizione esempio "Taranto è una provincia della Puglia" si possono fare degli esempi di predicati:
"X è provincia di Puglia" con X appartenente all'insieme delle città d'Italia.
"X è provincia di Y" con X appartenente all'insieme delle città d'Italia e Y appartenente all'insieme delle Regioni.

Se il valore di X è Roma, il primo predicato assumerà valore falso, se Y è Lazio il secondo predicato, alla presenza dello stesso valore di X, assumerà valore Vero.

Esercitazione:
4. Quali delle seguenti frasi di senso compiuto è una proposizione?
- Lorenzo Giovanotti è un cantante
- Non c'è festa più bella del Natale
- Oggi in classe ci sono più assenti di ieri
- Tre è pari
- Londra è la capitale del Brasile
- Il colore più bello è il viola
- Quelle scarpe sono costose
- Febbraio ha 30 giorni

5. Trasformate la proposizione "**Euro è la valuta dell'Italia**",
- in un predicato ad una variabile (R1)
- in un predicato a due variabili (R2)

Composizione di proposizioni semplici.

Le Proposizioni si dicono **Composte** quando sono formate da proposizioni semplici collegate da connettivi logici. Per esempio "la temperatura è zero gradi **E** sta nevicando" è una proposizione composta tramite una congiunzione logica (la **E** della frase che come vedremo tra poco corrisponde formalmente al connettivo logico **AND**), il valore di verità della proposizione composta dipenderà dai valori di verità delle proposizioni semplici e dal connettivo logico che li unisce, cioè nel caso che sia vero che la temperatura sia di zero gradi ma non sia vero che sta nevicando, la proposizione composta tramite la congiunzione logica ha valore di verità falso.

L'AND, la congiunzione logica.

Date due proposizioni P e Q l'operatore "AND" si può costruire la nuova proposizione "P AND Q" il cui valore di verità sarà VERO **solo se** i valori d'ingresso di P e di Q **sono entrambi veri**.

Il risultato della combinazione dei valori di verità delle proposizioni semplici connesse tramite un connettivo logico è descritto attraverso la Tavola di Verità, dove si assume di rappresentare **con 0 il valore FALSO e con 1 il valore VERO**.

P	Q	P AND Q
0	0	0
0	1	0
1	0	0
1	1	1

Esempio:

(Roma è la capitale di Francia) AND (2 è dispari) è Falsa perché 0 AND 0 = 0
(Roma è la capitale di Francia) AND (2 è pari) è Falsa perché 0 AND 1 = 0
(Roma è la capitale d'Italia) AND (2 è dispari) è Falsa perché 1 AND 0 = 0
(Roma è la capitale d'Italia) AND (2 è pari) è Vera perché 1 AND 1 = 1

L'ELEMENTO FORZANTE è quel valore d'ingresso che consente di definire il risultato dell'operatore binario senza dover necessariamente conoscere il valore dell'altro ingresso, nell'**AND** l'elemento forzante è lo **ZERO**, infatti, se il primo dei due valori d'ingresso è zero, si può escludere la possibilità di avere i due valori entrambi veri e quindi il valore d'uscita può essere solo Falso. Vale a dire:

0 AND X = 0;

L'ELEMENTO NEUTRO invece è quel valore d'ingresso che non consente di definire il risultato dell'operatore binario, anzi tale risultato sarà uguale al valore di verità dell'altro ingresso, nell'**AND** l'elemento neutro è l'**UNO**. Vale a dire:

1 AND X = X;

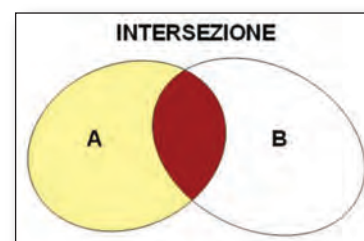
Se si prova in queste ultime due definizioni a sostituire l'AND con il "·", notazione spesso usata per tale operatore, otteniamo:

$$0 \cdot X = 0;$$
$$1 \cdot X = X;$$

cioè proprio il comportamento dell'operatore per della moltiplicazione, il che ci fa intuire perché l'AND viene anche chiamato **prodotto logico**.

Si può fare riferimento anche alla **teoria degli insiemi** affermando che l'operatore AND corrisponde all'operazione **d'intersezione**, per esempio se si devono selezionare per un corso di teatro pomeridiano tutti gli alunni di una scuola che frequentano il secondo anno e che sono stati promossi senza nessun debito formativo l'anno precedente, possiamo rappresentare graficamente la situazione in questo modo:

U = {insieme degli alunni della scuola},
A = {alunni della scuola che frequentano il secondo anno},
B = {alunni della scuola che sono stati promossi senza nessun debito formativo},
Intersezione = {alunni selezionati per il corso di teatro},

INTERSEZIONE
A
B

Cioè saranno selezionati tutti gli alunni che appartengono contemporaneamente all'insieme A e all'insieme B. Nella logica delle proposizioni avremmo affermato che X alunno della scuola sarà selezionato se e soltanto se risulta vero che:
(X frequenta il secondo anno) AND (X è stato promosso senza nessun debito formativo)

L'OR, la disgiunzione logica.

Date due proposizioni P e Q l'operatore "OR" si può costruire la nuova proposizione "P OR Q" per la quale sarà sufficiente che **almeno uno** dei valori d'ingresso sia **VERO** perché il valore d'uscita sia VERO.

P	Q	P OR Q
0	0	0
0	1	1
1	0	1
1	1	1

Esempio:

(Roma è la capitale di Francia) OR (2 è dispari) è Falsa perché 0 OR 0 = 0
(Roma è la capitale di Francia) OR (2 è pari) è Vera perché 0 OR 1 = 1
(Roma è la capitale d'Italia) OR (2 è dispari) è Vera perché 1 OR 0 = 1
(Roma è la capitale d'Italia) OR (2 è pari) è Vera perché 1 OR 1 = 1

L'ELEMENTO FORZANTE, quel valore d'ingresso che consente di definire il risultato dell'operatore binario senza dover necessariamente conoscere il valore dell'altro ingresso, nell'**OR** è l'**UNO**, infatti, se il primo dei due valori d'ingresso è UNO, si può affermare già che il valore d'uscita può essere solo VERO indipendentemente dall'altro ingresso. Vale a dire:

1 OR X = 1;

L'ELEMENTO NEUTRO, quel valore d'ingresso che non consente di definire il risultato dell'operatore binario ma tale risultato sarà uguale al valore di verità dell'altro ingresso, nell'**OR** l'elemento neutro è lo **ZERO**. Vale a dire:

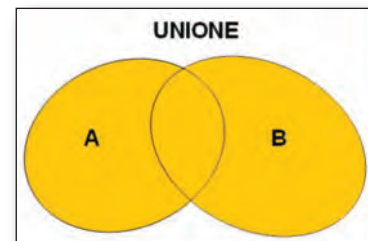
0 OR X = X;

Se si prova in queste ultime due definizioni a sostituire l'OR con il "+", notazione spesso usata per tale operatore, otteniamo:

$$0+X = X;$$

cioè proprio come nel comportamento dello zero nell'addizione, il che ci fa intuire perché l'OR viene anche chiamato somma logica.

Nella t**eoria degli insiemi** l'operatore OR corrisponde all'operazione di **unione**, per esempio se devo selezionare per un corso di teatro pomeridiano tutti gli alunni di una scuola che frequentano il secondo anno o che hanno già fatto recitazione saranno inclusi tutti quelli che frequentano il secondo anno, nessuno escluso, e tutti gli studenti che hanno frequentato un corso di recitazione, anche se sono di classi diverse dal secondo, il risultato corrisponde proprio all'unione dei due insiemi. Possiamo rappresentare graficamente la situazione in questo modo:

U = {insieme degli alunni della scuola},
A = {alunni della scuola che frequentano il secondo anno},
B = {alunni della scuola che hanno già frequentato un corso di recitazione},
Unione = { alunni selezionati per il corso di teatro},

Cioè saranno selezionati tutti gli alunni che appartengono all'insieme A oppure all'insieme B.
Nella logica delle proposizioni avremmo affermato che X alunno della scuola sarà selezionato se e soltanto se risulta vero che:

(X frequenta il secondo anno) OR (X ha già frequentato un corso di recitazione)

Il NOT, la negazione logica.

L'operatore logico NOT si differenzia dai precedenti AND e OR che sono detti operatori binari, cioè operano su due valori d'ingresso, mentre la negazione logica è un operatore **unario**, all'unico valore d'ingresso restituisce come valore d'uscita il suo opposto, per questo spesso si dice che si esegue il complemento del valore.

P	NOT P
0	1
1	0

Esempio:

NOT (Roma è la capitale di Francia) è vero perché NOT(0) = 1

NOT (Roma è la capitale d'Italia) è falso perché NOT(1) = 0

Nella **teoria degli insiemi** l'operatore NOT corrisponde all'insieme complemento, per esempio se devono partecipare a una selezione per un corso di teatro pomeridiano tutti gli alunni di una scuola che NON frequentano il quinto anno cioè che frequenta qualsiasi classe tranne la quinta, il risultato si ottiene partendo dall'insieme degli alunni che frequenta una quinta classe e facendone l'operazione di complemento. Possiamo rappresentare graficamente la situazione in questo modo:

U = {insieme degli alunni della scuola},
A = {alunni della scuola che frequentano la quinta classe},
Complemento ={alunni che partecipano alla selezione per il corso di teatro},

Cioè saranno selezionati tutti gli alunni che NON appartengono all'insieme A. Nella logica delle proposizioni avremmo affermato che X alunno della scuola sarà selezionato se e soltanto se risulta vero che:

NOT (X frequenta la quinta classe)

Uno sguardo in Rete:
La Boolean Machine: http://rockwellschrock.com/rbs3k/boolean/index.htm
Un viaggio nell'algebra di Boole
http://www.labsquare.it/index.php/categorie/mediazone/20-un-viaggio-nell-algebra-di-boole
Learning Objects: Università di Cassino, Laboratorio di Tecnologie della Conoscenza e dell'educazione (http://tle.let.unicas.it/LO_index.html)
La logica delle proposizioni e il linguaggio: Piccola guida con quiz di verifica dell'apprendimento, per l'utilizzo si consiglia di eseguire il download dal link http://tle.let.unicas.it/LOs/log-prop_WEB.zip, decomprimere in una cartella locale e aprire il file index.html con un qualsiasi programma di navigazione web, il contenuto è quindi utilizzabile off-line (il materiale presente nel sito è sottoposto ad una licenza di tipo Creative Commons)

Una semplice dimostrazione di come funzionano i connettivi AND e OR si può fare per analogia su un circuito elettrico, come mostrato nel video collegato al codice QR o al link equivalente.

AND e OR
http://qrbridge.me/9n80

Simbolismi e Significati

L'algebra di Boole e i connettivi logici sono anche alla base dell'elettronica digitale, una porta logica è un circuito digitale in grado di eseguire la funzione logica di un connettivo logico su una o più variabili booleane. Nella tabella seguente si rappresentano i simboli logici equivalenti in diversi simbolismi.

Nome operatore	Logica delle proposizioni	Altre notazioni utilizzate	Elettronica Digitale	Dal latino
Negazione logica	NOT(A)	A, ~A, ¬A,!A		non
Congiunzione	A AND B	A·B, A ∧ B, A∩B		et
Disgiunzione inclusiva	A OR B	A+B, A ∨ B, AUB		vel

Esercitazione:
Date le proposizioni: **P =“Leo viene in autobus”**; **Q =“Giorgia viene in bicicletta”** trasforma le proposizioni composte rappresentandole con le variabili P e Q usando i connettivi logici richiesti:

Es: Leo viene in autobus o Giorgia viene in bicicletta: **P Or Q**;
1. Leo viene in autobus e Giorgia non viene in bicicletta;
2. Leo non viene in autobus o Giorgia viene in bicicletta;
3. Leo non viene in autobus e Giorgia non viene in bicicletta;
4. Leo viene in autobus e Giorgia viene in bicicletta;

Risolvere e motivare le seguenti espressioni:

- 1 OR X = _______ perché______________________________
- X AND 0 = _______ perché______________________________
- X OR _______ = X perché ______________________________
- 1 AND______ = X perché ______________________________
- X OR NOT(X) = _______ perché ______________________________
- X AND NOT(X) = _______ perchè ______________________________

Espressioni Booleane

L'**Espressione Booleana** o **Funzioni Logica** di k Variabili d'ingresso è una procedura che permette di associare univocamente a ogni **configurazione di valori** di verità attribuiti alle Variabili K un valore di verità in uscita.

Si osservi che quando abbiamo definito la tavola di verità dell'AND in realtà abbiamo calcolato il Valore di Verità dell'espressione booleana (**P AND Q**) in uscita per ogni **configurazione di valori** di verità delle due variabili d'ingresso **P** e **Q**, ovviamente le combinazioni possibili sono quattro cioè 2^k, dove **k** rappresenta il numero di variabili mentre 2 è la base perché i simboli utilizzati sono solo 2 cioè 0 e 1.

P	Q	P AND Q
0	0	0
0	1	0
1	0	0
1	1	1

Calcoliamo ad esempio la funzione logica f: (P,Q) = NOT(P AND Q):

P	Q	P AND Q	NOT (P AND Q)
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Si osservi che l'espressione prevede di applicare la negazione logica al risultato dell'AND quindi l'ultima colonna contiene sempre il complemento binario, cioè il valore opposto del corrispondente valore presente nella terza colonna.

Calcoliamo la funzione logica **f: (P,Q) = (NOT(P AND Q)) OR ((P AND NOT(Q))**

Per risolvere questo esercizio è necessario procedere per fasi successive:
fase 1: Calcolare i valori di verità per (NOT(P AND Q))
fase 2: Calcolare i valori di verità per (P AND NOT (Q))
fase 3: Applicare il connettivo OR ai risultati ottenuti nella fase 1 e 2

fase 1				fase 2		fase 3
P	Q	P AND Q	NOR (P AND Q)	NOT (Q)	P AND NOT (Q)	NOT (P AND Q) OR P AND NOT (Q)
0	0	0	1	1	0	1
0	1	0	1	0	0	1
1	0	0	1	1	1	1
1	1	1	0	0	0	0

Possiamo affermare che per l'esercizio svolto la funzione f è sempre vera tranne quando i valori d'ingresso sono P e Q sono entrambi veri.
Calcoliamo la funzione logica **f: (X,Y,Z) = (NOT(X AND Y)) OR Z**

In questa funzione le variabili sono tre, quindi **le combinazioni d'ingresso diventano 2^3=8**, è opportuno scegliere una regola comune nel disporre le variabili nella tabella e nella costruzione delle combinazioni possibili, il metodo più usato è quello che è stato utilizzato nella seguente risoluzione dell'esercizio:

X	Y	Z	X AND Y	NOT (X AND Y)	(NOT (X AND Y) OR Z
0	0	0	0	1	1
0	0	1	0	1	1
0	1	0	0	1	1
0	1	1	0	1	1
1	0	0	0	1	1
1	0	1	0	1	1
1	1	0	1	0	0
1	1	1	1	0	1

La Risoluzione per sostituzione è un'altra modalità di esercizio e risoluzione delle espressioni booleane, consiste nel sostituire le variabili con i valori definiti nella traccia, per esempio si esegua il seguente esercizio in cui prendiamo come funzione quella appena risolta con la tabella nell'esercizio precedente:

Dati i **valori d'ingresso (X=1; Y=0; Z=0)** si calcoli l'uscita dell'espressione booleana f:(X,Y,Z) = (NOT(X AND Y)) OR Z per sostituzione;

svolgimento:

1. si sostituiscono nella funzione f:(X,Y,Z)=(NOT(X AND Y))OR Z i valori (X=1; Y=0; Z=0)
2. f(1,0,0) = (NOT(1 AND 0)) OR 0; applicando la tavola di verità dell'AND otteniamo
3. f(1,0,0) = (NOT(0)) OR 0; applicando la tavola di verità del NOT otteniamo
4. f(1,0,0) = 1 OR 0; applicando la tavola di verità dell'OR otteniamo
5. f(1,0,0) = 1

Se si confronta lo svolgimento dell'esercizio con quello della stessa funzione svolto in precedenza con la tabella, nella riga in corrispondenza della configurazione d'ingresso (X=1; Y=0; Z=0) il risultato dell'espressione coincide con quello ottenuto nella risoluzione per sostituzione.

Risolvendo un'espressione logica o funzione booleana può verificarsi che le uscite siano sempre vere o sempre false, esiste una definizione per tali tipologie di espressioni.
Una funzione logica che per qualsiasi combinazione di valori delle variabili risulta sempre vera si definisce una **tautologia**, una funzione logica che per qualsiasi combinazione di valori delle variabili risulta sempre falsa si definisce una **contraddizione**.

Ordini di priorità dei connettivi logici

Nelle espressioni logiche non è sempre necessario usare le parentesi ma si possono risolvere applicando gli ordini di priorità, dalla priorità più alta alla più bassa l'ordine è:

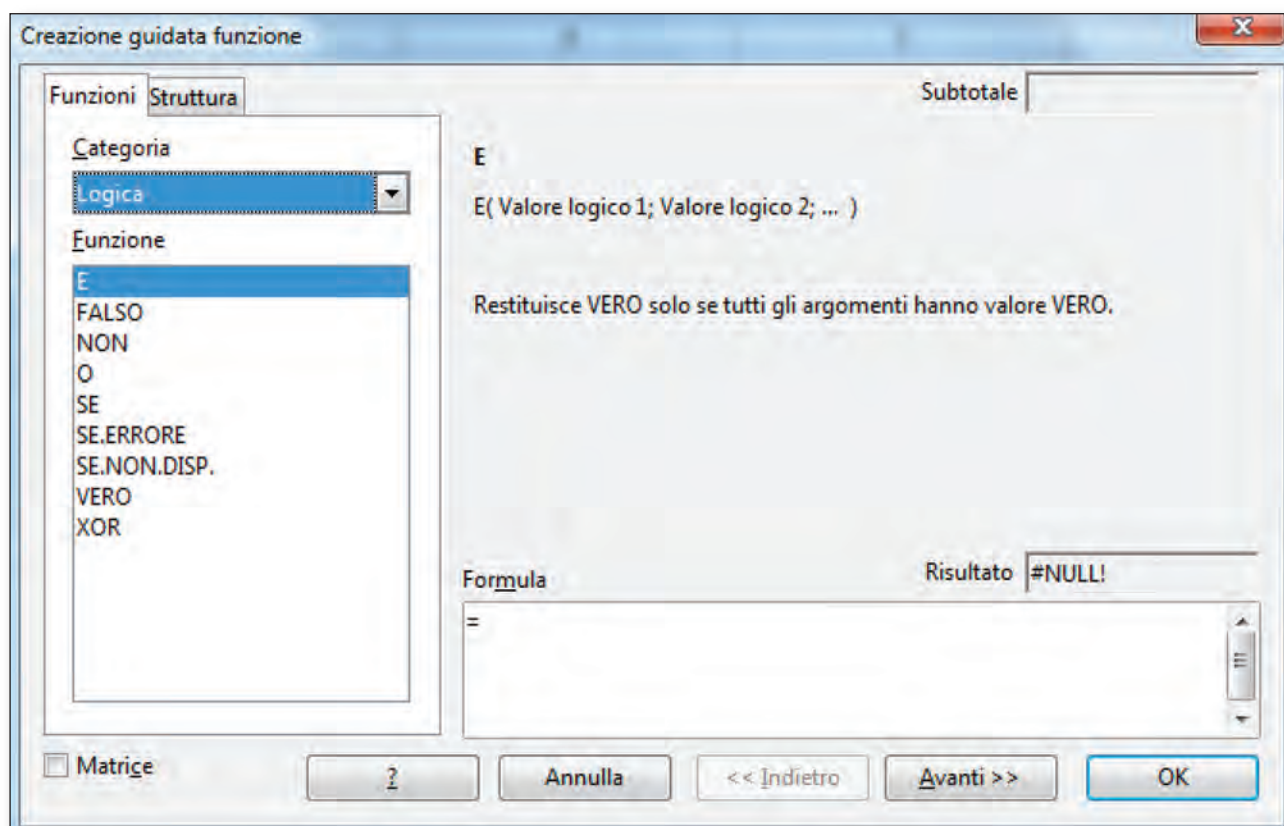
NOT, AND, OR.

Se osserviamo la seguente espressione aritmetica: **2+3·5** sappiamo che equivale a scrivere **2+(3·5)**, il prodotto è prioritario sulla somma, analogamente nell'espressione **A OR B AND C** dobbiamo considerare il prodotto logico AND prioritario rispetto alla somma logica OR quindi l'espressione va risolta come se ci fossero le parentesi tonde **A OR (B AND C)**.

Il **NOT** ha priorità sugli altri operatori logici, quindi l'espressione **A OR NOT B AND C** si esegue come se ci fossero le parentesi in questo modo **A OR ((NOT B) AND C)**.

Gli operatori logici nei fogli elettronici

I fogli elettronici prevedono tra le categorie di funzioni le Funzioni Logiche e tra queste gli operatori booleani principali che abbiamo introdotto in questo capitolo. Le applicazioni per il foglio elettronico più utilizzate sono Microsoft® Excel e Calc di Apache OpenOffice™ o di LibreOffice, il nome delle funzioni sono le stesse, così come la loro sintassi, pertanto l'esempio che segue, nel quale useremo LibreOffice Calc ver. 4.1, può essere eseguito con una qualsiasi delle applicazioni citate.

Se dopo aver aperto l'applicazione LibreOffice Calc ver. 4.1 clicchiamo su Inserisci – Funzione, appare la finestra di creazione guidata funzione, selezioniamo nel menù a discesa di Categoria la voce Logica, apparirà nel menù Funzione sottostante l'elenco delle funzioni logiche disponibili.

Possiamo facilmente individuare nell'elenco le funzioni E, O, NON, che corrispondono rispettivamente agli operatori logici AND, OR, NOT; probabilmente in questo caso, sarebbe stato opportuno non tradurre per le versioni italiane i nomi delle funzioni, ma bisogna comprendere che questi strumenti sono progettati per l'utilizzo da parte di utenti che non necessariamente hanno studiato l'informatica e che quindi potrebbero non aver mai sentito parlare degli operatori booleani.

L'operatore E: sintassi E(valore logico1;Valore logico2;…)

Restituisce Vero solo se tutti gli argomenti hanno valore Vero, notiamo quindi che nel foglio elettronico è possibile calcolare la funzione su più argomenti, ma non esamineremo casi simili non utili ai fini del nostro obiettivo che è la risoluzione di espressioni booleane con il foglio elettronico.

La funzione A AND B quindi diventerebbe E(A;B),

	A	B	C	D	E	F	G	H
1	A	B	A AND B			A	B	A AND B
2	0	0	FALSO			FALSO	FALSO	FALSO
3	0	1	FALSO			FALSO	VERO	FALSO
4	1	0	FALSO			VERO	FALSO	FALSO
5	1	1	VERO			VERO	VERO	VERO

È importante notare che i valori accettati in ingresso sono sia 1 o 0 sia gli analoghi VERO o FALSO, mentre la funzione mostra come risultato sempre i valori VERO o FALSO.

L'operatore O: sintassi O(valore logico1;Valore logico2;…)
Restituisce Vero quando anche solo uno degli argomenti ha valore Vero, la funzione A OR B che diventerebbe O(A;B)

A	B	A OR B
0	0	FALSO
0	1	VERO
1	0	VERO
1	1	VERO

Infine l'operatore NON: sintassi NON(valore logico)
Restituisce il valore opposto a quello dell'unico argomento ha valore Vero, la funzione NOT(A) diventerebbe NON(A)

A	NOT A
0	VERO
1	FALSO

Risolviamo l'espressione booleana: **F(P,Q): (P or (Not (Q)) or (Q and not(P))**

F(P,Q): (P or (Not (Q)) or (Q and not(P))

P	Q	not P	not Q	P or Not Q	Q and not P	F(P,Q)
0	0	VERO	VERO	VERO	FALSO	VERO
0	1	VERO	FALSO	FALSO	VERO	VERO
1	0	FALSO	VERO	VERO	FALSO	VERO
1	1	FALSO	FALSO	VERO	FALSO	VERO

Sono stati inseriti solo alcuni commenti alle celle evidenziando qual è la formula corrispondente, si può osservare che l'espressione booleana è una tautologia perché sempre vera per qualsiasi combinazione di valori di P e Q in ingresso.

Volendo non è necessario risolvere l'espressione booleana per parti successive, ma si può costruire un'unica formula sfruttando la tecnica dell'annidamento, cioè l'inserimento di una formula come argomento di un'altra. Vediamo l'uso di tale tecnica nella figura successiva, dove si risolve l'espressione booleana con tre variabili d'ingresso:

f(A,B,C): (A or not C) and (B or not A)

	A	B	C	D	E	F
1			(A or not C) and (B or not A)			
2	A	B	C	=E(O(A3;NON(C3));O(B3;NON((A3))))		
3	0	0	0		VERO	
4	0	0	1		FALSO	
5	0	1	0		VERO	
6	0	1	1		FALSO	
7	1	0	0		FALSO	
8	1	0	1		FALSO	
9	1	1	0		VERO	
10	1	1	1		VERO	

Esercizi:

1. Risolvere le espressioni booleane
   f:(X,Y) = (X and not Y) or (not X and not Y)
   f:(X,Y,Z) = (Z or not X) and (Y or not X)
2. Risolvere per sostituzione f:(X,Y) = (X and not Y) or (Y and not X) con X=1 e Y = 0
3. Risolvere sul foglio elettronico l'espressione booleana inserita nella riga 1

	A	B	C	D	E	F
1	F(A,B): (NOT B OR NOT A AND B)					
2						
3	A	B				
4	0	0				
5	0	1				
6	1	0				
7	1	1				

Approfondimento:

Proprietà e Teoremi: In questa sezione riportiamo alcune proprietà degli operatori logici ed alcuni teoremi fondamentali delle Algebre di Boole.

Proprietà Commutativa: scambiando il valore di X con quello di Y il valore di verità del risultato non cambia

X AND Y = Y AND X il prodotto logico gode della proprietà commutativa, col simbolo "·":

$$X \cdot Y = Y \cdot X$$

X OR Y = Y OR X la somma logica gode della proprietà commutativa, col simbolo "+":

$$X + Y = Y + X$$

Proprietà Distributiva del prodotto logico rispetto alla somma logica

X AND (Y OR Z) = X AND Y OR X AND Z con i simboli "·" e "+" equivale a dire: $X \cdot (Y+Z) = X \cdot Y + X \cdot Z$

(si suggerisce di dimostrare il teorema risolvendo prima la funzione logica a sinistra dell'= poi quella a destra e confrontare i valori di verità).

Primo teorema dell'assorbimento: X+X·Y= X

Dimostrazione:

Si mette X a fattore comune X(1+Y)=X ma 1+Y=1 quindi X·1=X quindi X=X che è sempre vero.

Secondo Teorema dell'assorbimento: X+NOT(X)·Y= X+Y

(si suggerisce di dimostrare il teorema risolvendo prima la funzione logica a sinistra del simbolo "=" poi quella a destra e confrontare i valori di verità).

Primo teorema di De Morgan: NOT(X+Y)= NOT(X)·NOT(Y)

(si suggerisce di dimostrare il teorema risolvendo prima la funzione logica a sinistra del simbolo "=" poi quella a destra e confrontare i valori di verità).

Secondo teorema di De Morgan: NOT(X·Y)= NOT(X)+NOT(Y)

(si suggerisce di dimostrare il teorema risolvendo prima la funzione logica a sinistra del simbolo "=" poi quella a destra e confrontare i valori di verità).

Teorema fondamentale di De Morgan:

Afferma che per calcolare la **Funzione negata (Not F)** basta negare tutte le variabili presenti nella funzione e sostituire l'operatore somma (OR) con l'operatore prodotto (AND) e viceversa.

Questo teorema viene utilizzato per ottenere una diversa rappresentazione della funzione logica, risulta particolarmente utile per passare dalla rappresentazione di funzioni espresse come somme di prodotti logici, dette rappresentazioni in **prima forma canonica**, alla rappresentazione di funzioni espresse come prodotti di somme, dette rappresentazioni in **seconda forma canonica**,

per esempio

data F = (NOT(X)· NOT (Y)) + (X·Y) (espressione in prima forma canonica)

allora Not F = (X+Y) · (NOT(X)+NOT (Y)) (espressione in seconda forma canonica)

Ogni elemento della sommatoria nella prima forma canonica viene chiamato **minterm** o termine elementare.

Ogni fattore del prodotto nella seconda forma canonica viene chiamato **maxterm** o termine massimo.

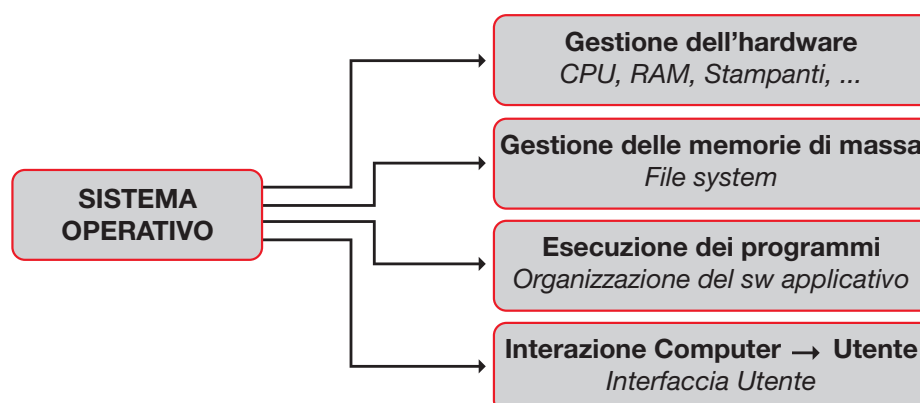
Questo teorema viene spesso utilizzato insieme agli altri teoremi, quindi con un **metodo algebrico**, per la semplificazioni delle espressioni logiche, ricordiamo che esistono altri metodi, cioè le **Mappe di Karnaugh, il Metodo di Quine-McClusky e la Fattorizzazione**.

4. Il Sistema Operativo

Autore: **Luca Peresson**
rielaborazione a cura di **Dario Rinaudo** e **Angelo Oliva**

Competenze	Abilità	Conoscenze
Utilizzare, con autonomia e responsabilità, le reti e gli strumenti informatici0 nelle attività di studio, ricerca e approfondimento disciplinare.	*Riconoscere le caratteristiche logico-funzionali di un computer e il ruolo strumentale svolto nei vari ambiti.* *Tipologie ed evoluzioni di un Sistema Operativo.*	*Struttura e funzioni di un sistema operativo Interfacce e periferiche di un computer*

Il Sistema Operativo

Con il termine *Sistema Operativo* (o talvolta *software di base*) si indica quell'insieme di programmi destinato -genericamente- alla gestione del computer. Tra le principali funzioni da esso svolto troviamo la gestione dell'hardware, la memorizzazione dei dati nelle memorie di massa, l'esecuzione dei programmi applicativi e l'interazione fra il computer e l'utente.

I diversi sistemi operativi

Nel mercato mondiale esistono numerosi sistemi operativi che possono essere classificati secondo diversi criteri:

– per famiglia
seguendo questa classificazione possiamo individuare i sistemi della famiglia ***Unix*** e ***Microsoft***. Alla prima famiglia appartengono, fra gli altri, i sistemi operativi di tipo *Linux* (nelle diverse "distribuzioni" quali *Slackware, Fedora, Ubuntu, Debian, Suse, Gentoo, Mandriva*,...), *BSD* (tra i quali è d'obbligo citare *FreeBSD*, *OpenBSD* e *DarwinOS*) e gli stessi sistemi *Unix*;. Alla famiglia Microsoft appartengono i sistemi operativi XENIX, MSDOS, OS/2 e la serie di sistemi operativi di tipo *Windows* (tra i quali le recenti *XP, Vista, Server 2008, Mobile, 7, …*);

– per tipo di licenza
seguendo questa classificazione si individuano i cosiddetti sistemi proprietari e i sistemi open source;

– *per dispositivo*

è una classificazione che tiene conto del tipo di elaboratore sul quale è destinato a funzionare il sistema operativo. In questo caso potremo individuare sistemi operativi orientati alla gestione di *smartphone* (computer integrati a dispositivi telefonici), di lettori di *eBook* (dispositivi portatili orientati alla lettura di documenti in formato elettronico), di *personal computer* (computer da tavolo o portatili) fino alla gestione di *mainframe* (detti anche *sistemi centrali*, gli elaboratori considerati di alta fascia e destinati alla gestione di applicazioni particolarmente delicate che necessitano di alta affidabilità);

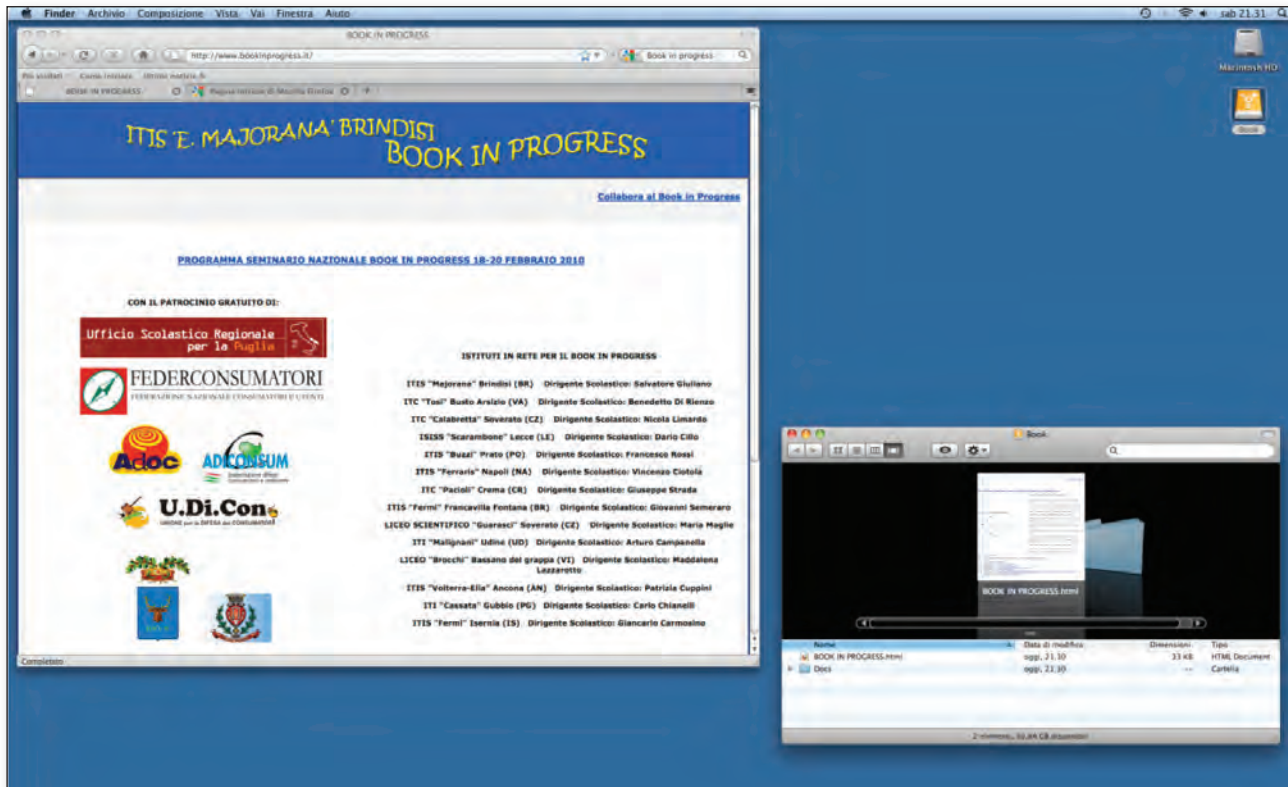
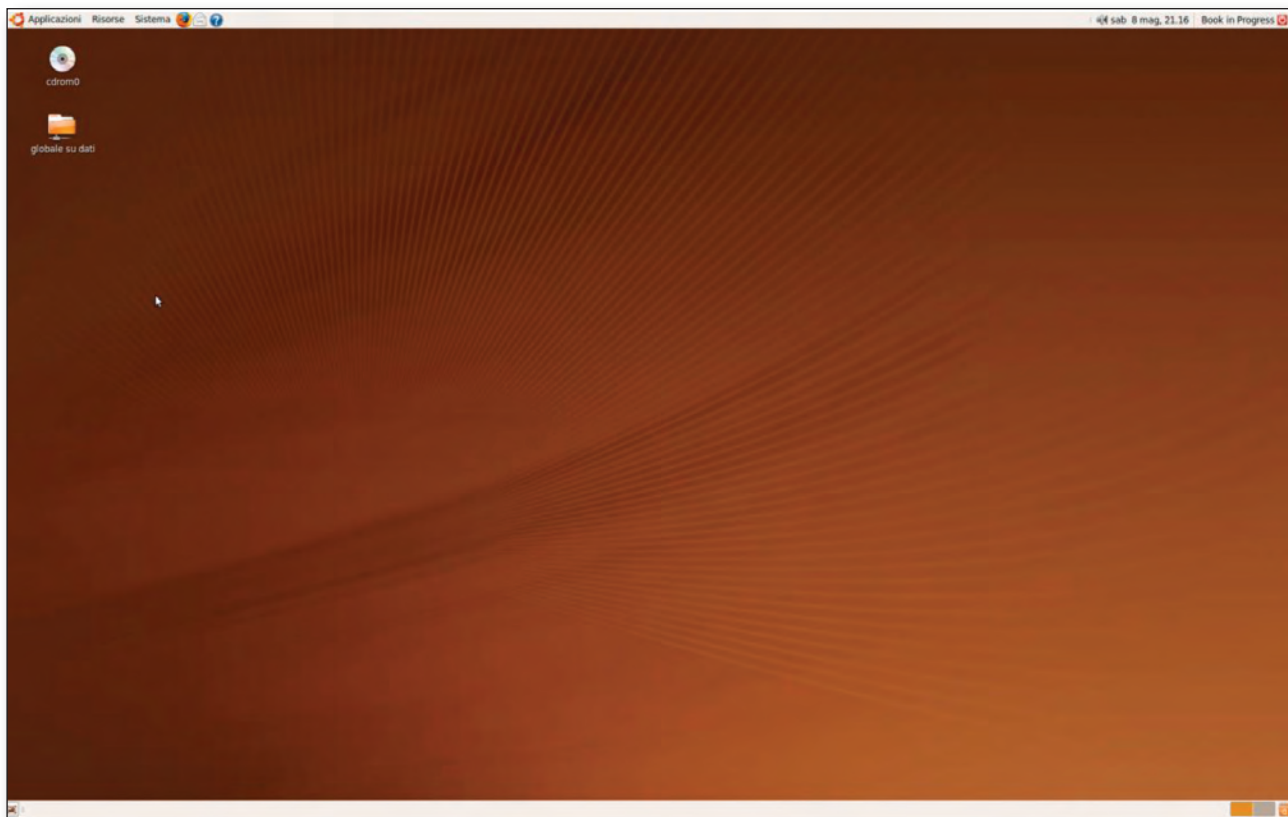
– *per applicativo*

è una classificazione che tiene conto dei programmi che verranno utilizzati dal sistema. In questo caso potremo individuare sistemi operativi in grado di eseguire particolari e specifici programmi orientati a specifici settori quali la grafica, l'office automation, l'amministrazione/gestione aziendale, il controllo automatico di processi produttivi, l'intrattenimento, ecc..

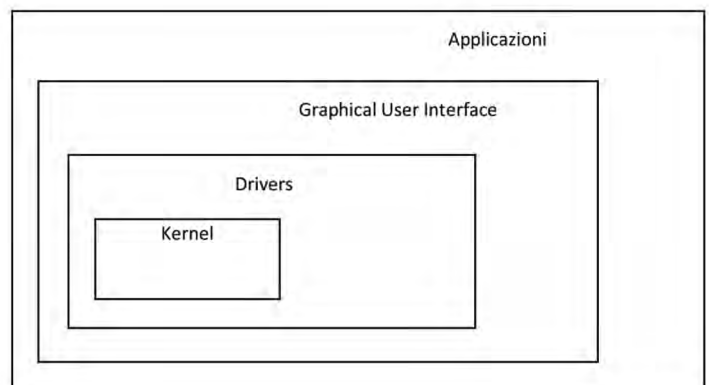
La struttura di un sistema operativo

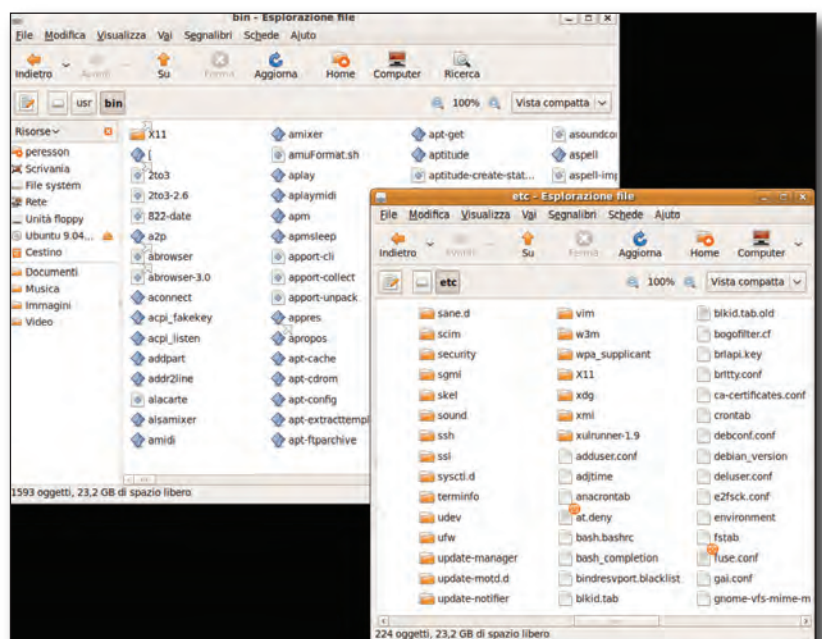
Nell'immaginario legato all'uso del computer, un sistema operativo viene individuato dal tipo di gestione del monitor e del mouse, di quel programma che tecnicamente viene indicato con il termine *interfaccia grafica*. Siamo quindi abituati ad associare un dato sistema operativo a come esso si presenta graficamente al suo avvio.

Questa semplificazione è perfettamente giustificata dal fatto che è l'innovazione rappresentata dall'uso del mouse e delle finestre ad aver permesso la straordinaria diffusione dei computer in questi ultimi decenni.

Poter “aprire” delle *finestre*, essere in grado di visualizzare il contenuto di una memoria di massa attraverso la sua rappresentazione in *icone* e *cartelle* ha semplificato l'organizzazione dei contenuti nei supporti di memorizzazione; poter avviare dei programmi attraverso un *doppio clic* del tasto di un mouse o selezionare aree di un documento attraverso l'azione di *trascinamento* ha consentito di velocizzare l'accesso e l'utilizzo dei programmi applicativi.

Tuttavia, il software di interfaccia grafica non rappresenta nemmeno la millesima parte dei moduli che vengono controllati e gestiti da un sistema operativo. E solo la componente delsistema operativo più “vicina” all'uomo. Non l'unica e certamente non la più necessaria al suo funzionamento.

Volendo schematizzare il sistema operativo nelle sue componenti essenziali possiamo individuare:

– il kernel, o nucleo

che è l'insieme dei programmi destinati a gestire l'accesso di tutti gli altri programmi all'hardware della macchina. L'efficienza di questo modulo determina la velocità con cui i programmi vengono eseguiti considerato che è questo modulo ad assegnare ai programmi l'accesso a risorse quali il processore e la memoria centrale;

– il file system

che è l'insieme di regole che stabi-liscono come viene identificato un *file* all'interno dei dispositivi di memorizzazione di un computer. Il concetto di *file system* (e, quindi di *file)* è uno dei concetti più impor-tanti e delicati dell' informatica, basti pensare che dagli stessi dispositivi periferici (tastiera, stampante) alle risorse di rete (sistemi remoti, pagine html) viene associata una rappresentazione basata sul modello del file system;

– *il memory management;*

il gestore della memoria è quella parte di sistema operativo destinato a gestire la memoria centrale. Lo scopo di questo modulo è quello di assegnare ad ogni processo una porzione di memoria centrale facendo attenzione a che il funzionamento di un dato programma non vada a occupare zone di memorie già utilizzate da altri programmi

– *la memoria virtuale*

è una è una complessa tecnica di gestione della memoria che riguarda l'estensione dello spazio disponibile nella RAM attraverso l'utilizzo di una parte dello spazio di memoria dell'hard disk.
Considerata la significativa differenza di capacità e di velocità fra i due dispositivi il migliore bilanciamento fra gestione dello scambio dei dati e l'utilizzo delle risorse permette di “simulare” una memoria centrale significativamente maggiore di quella disponibile senza una sensibile diminuzione delle prestazioni della macchina;

– *i driver o device driver*

sono quei moduli software che consentono al sistema operativo di gestire un determinato dispositivo fisico attraverso l'uso delle interfacce standard. L'utilizzo di questi moduli è di estrema importanza nel funzionamento di un elaboratore perché permette da un lato la semplificazione della gestione delle periferiche e dall'altro un notevole aumento della flessibilità nella configurazione e nell'aggiornamento della configurazione hardware del sistema;

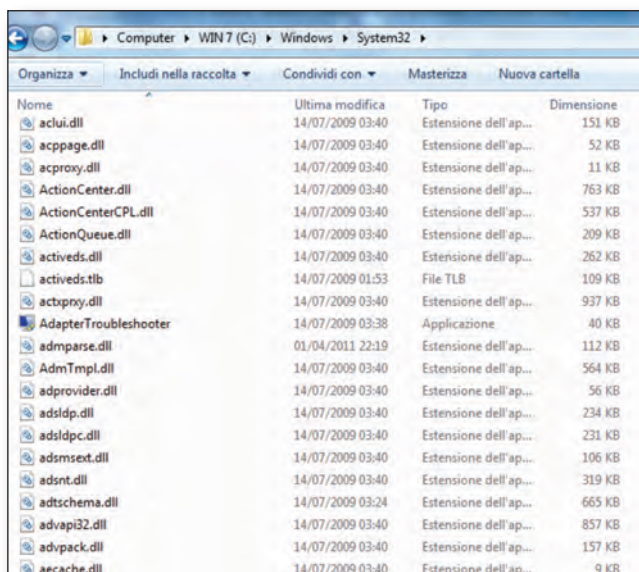
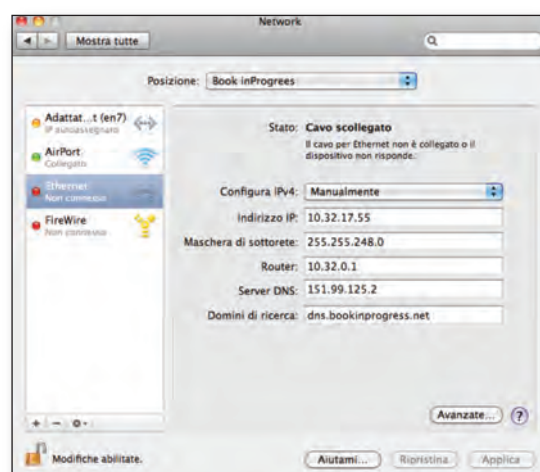
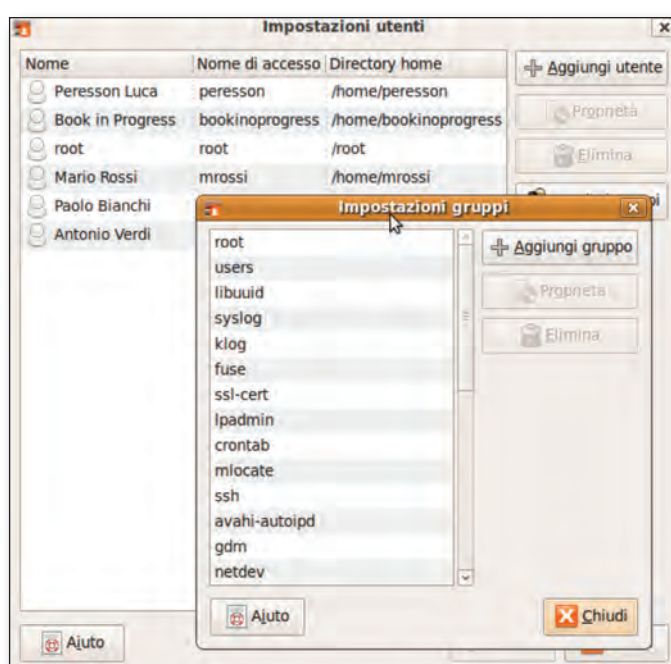
– *il multitasking*

concetto ormai acquisito da molte generazioni di elaboratori, è la capacità di un sistema di eseguire contemporaneamente diversi programmi assegnando ad ognuno di questi l'utilizzo delle stesse risorse (CPU, RAM, hard disk, monitor, ecc.);

– *Il gestore delle connessioni di rete*

conferisce al sistema la capacità di effettuare uno scambio di dati attraverso le infrastrutture di rete con altri sistemi. Tale funzione è legata all'utilizzo di diversi *protocolli* di rete ed alla capacità di supportare i programmi applicativi e i processi ad essi correlati;

– *la gestione di utenti*

è la caratteristica di un sistema di creare registrare e organizzare diversi metodi di accesso che garantiscano una graduale scala di priorità e un controllato utilizzo alle proprie risorse. A questa caratteristica è fortemente legato il tema della sicurezza e della vulnerabilità del sistema ad attacchi esterni;

I Driver

Connessioni di rete

Impostazioni Utenti

– l'interfaccia grafica o GUI (Graphical User Interface)

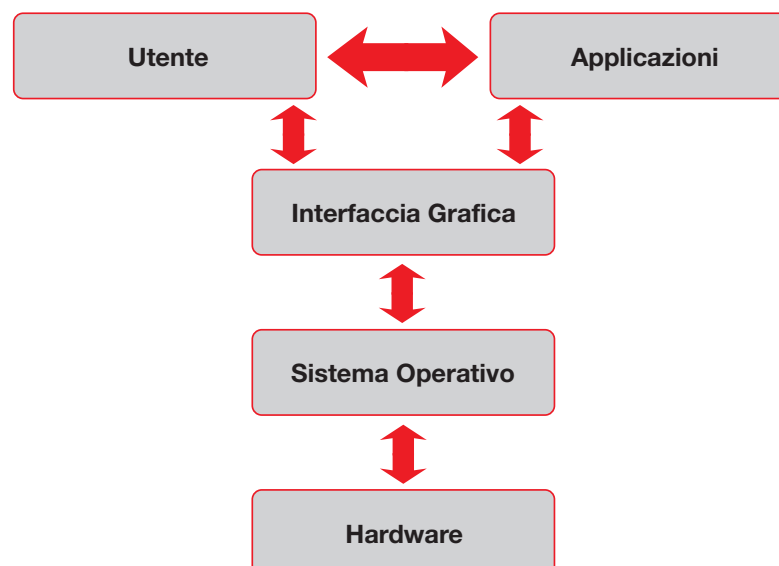
è quel software che consente all'utente di inviare dei comandi all' elaboratore facendo uso di dispositivi fisici (mouse e tastiera) che gestiscono oggetti grafici bidimensionali (puntatore, icone, finestre). In tal modo l'utente viene liberato dall' obbligo di utilizzare la sola tastiera per la scrittura e l'invio di comandi testuali (la più antica forma di utilizzo del sistema: la CUI, Character User Interface ossia interfaccia utente basata sul carattere);

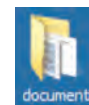
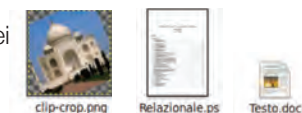
```
C:\Windows\system32\cmd.exe
icrosoft Windows [Versione 6.1.7600]
opyright (c) 2009 Microsoft Corporation. Tutti i diritti riservati.

:\Users\SALVATORE MADARO>dir
Il volume nell'unità C è WIN 7
Numero di serie del volume: E369-8C80

Directory di C:\Users\SALVATORE MADARO

1/04/2011  20:40    <DIR>          .
1/04/2011  20:40    <DIR>          ..
1/04/2011  21:30    <DIR>          Contacts
7/04/2011  20:10    <DIR>          Desktop
1/04/2011  21:30    <DIR>          Documents
7/04/2011  21:02    <DIR>          Downloads
1/04/2011  21:30    <DIR>          Favorites
1/04/2011  21:30    <DIR>          Links
1/04/2011  21:30    <DIR>          Music
1/04/2011  21:30    <DIR>          Pictures
1/04/2011  21:30    <DIR>          Saved Games
1/04/2011  21:30    <DIR>          Searches
1/04/2011  21:30    <DIR>          Videos
               0 File              0 byte
              13 Directory  34.847.920.128 byte disponibili

:\Users\SALVATORE MADARO>*.doc
*.doc" non è riconosciuto come comando interno o esterno,
un programma eseguibile o un file batch.

:\Users\SALVATORE MADARO>time
ra corrente: 23:38:27.72
mmettere nuova ora:

:\Users\SALVATORE MADARO>ipconfig

onfigurazione IP di Windows


cheda LAN wireless Connessione rete wireless:

  Suffisso DNS specifico per connessione: homenet.telecomitalia.it
  Indirizzo IPv6 locale rispetto al collegamento . : fe80::742c:5e03:334d:5660%
1
  Indirizzo IPv4. . . . . . . . . . . . : 192.168.1.238
  Subnet mask . . . . . . . . . . . . . : 255.255.255.0
  Gateway predefinito . . . . . . . . . : 192.168.1.1

cheda Ethernet Connessione alla rete locale (LAN):

  Stato supporto. . . . . . . . . . . . : Supporto disconnesso
  Suffisso DNS specifico per connessione: homenet.telecomitalia.it

cheda Tunnel Connessione alla rete locale (LAN)* 2:

  Suffisso DNS specifico per connessione:
  Indirizzo IPv6 . . . . . . . . . . . . . . . . : 2001:0:4137:9e76:3458:194d
a011:ecd2
```

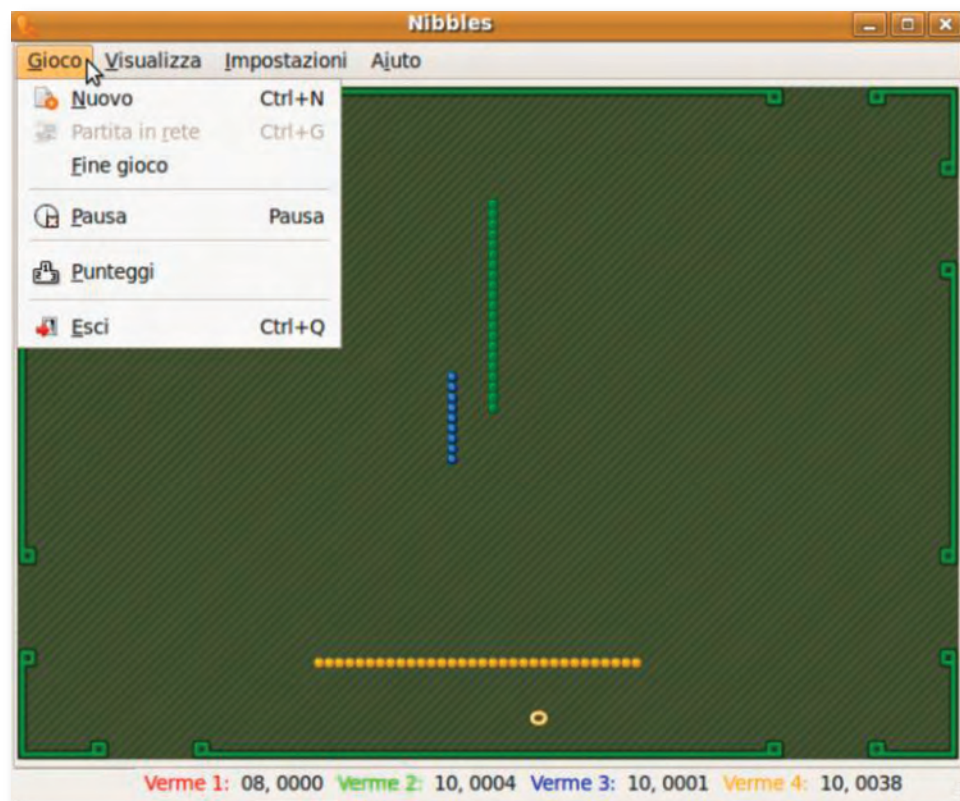
Graphical User Interface (GUI)

L'utilizzo dell'interfaccia grafica

Il funzionamento dell'interfaccia grafica si deve principalmente all'invenzione di un piccolo dispositivo di puntamento chiamato *mouse*, che consente, attraverso il proprio spostamento, di muovere un *puntatore* (un'immagine, generalmente una freccia, talvolta indicata con il termine *cursore*) sullo schermo e -attraverso la pressione su di uno o più pulsanti- di eseguire particolari operazioni.

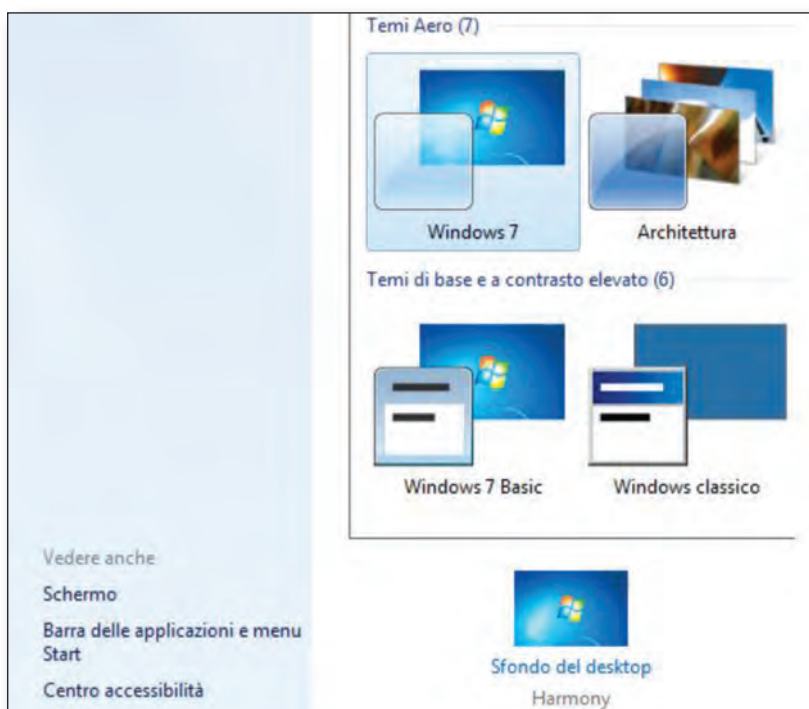
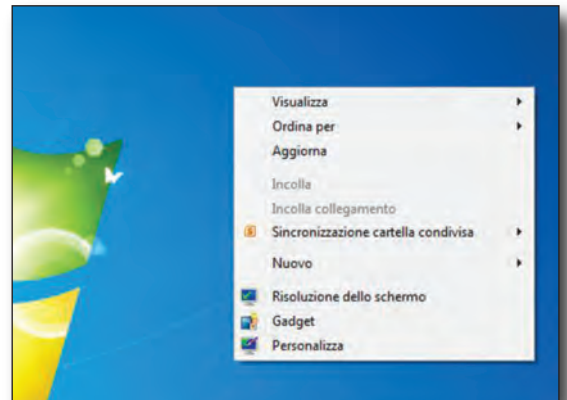
Questo semplice e rivoluzionario dispositivo viene ideato dall'inventore statunitense Douglas Carl Engelbart e realizzato come prototipo nel 1963. Solo dopo vent'anni, agli inizi degli anni '80 il dispositivo inizia a diffondersi e acquisisce notorietà nel 1984 con il computer *Apple Macintosh*, che presenta una semplice interfaccia grafica il cui funzionamento si basava su finestre e icone. Attualmente ogni sistema che preveda una frequente interazione di tipo *uomo-macchina* (tipicamente i sistemi installati su personal computer) rende disponibile uno o più tipi di interfaccia grafica.

Le più diffuse interfacce grafiche sono, come si è già accennato, la *Microsoft Windows*, la *Mac Os X* e la *X Windows System*. Ognuna di queste presenta uno, stilisticamente diverso, ambiente desktop che simula un piano di lavoro (definito scrivania) sul quale vengono disposti alcuni oggetti grafici:

- ***la cartella***
  rappresentazione grafica della "directory", l'area di memoria destinata all'archiviazione di documenti e programmi;

- ***i documenti***
  rappresentazione grafica dei "file di dati", la struttura di dati destinata alla memorizzazione dei risultati di un'elaborazione;

- ***i programmi***
  rappresentazione grafica dei "file eseguibili", la struttura di dati destinata alla memorizzazione dei programmi;

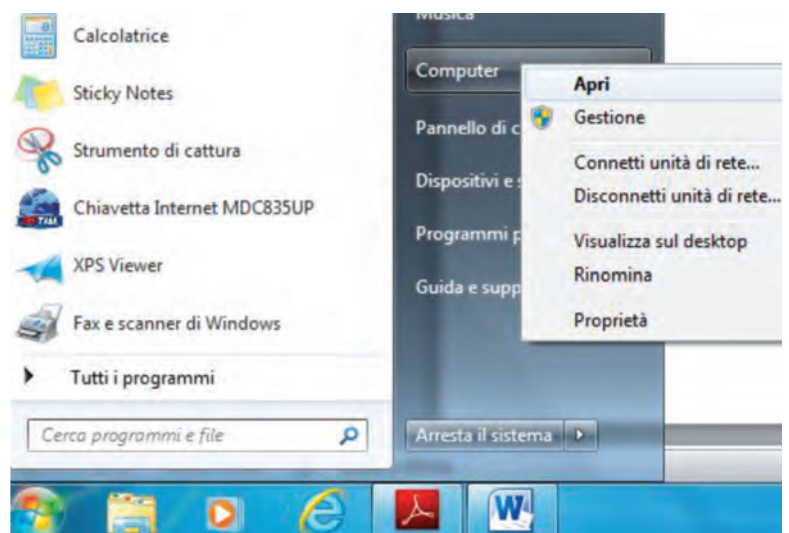
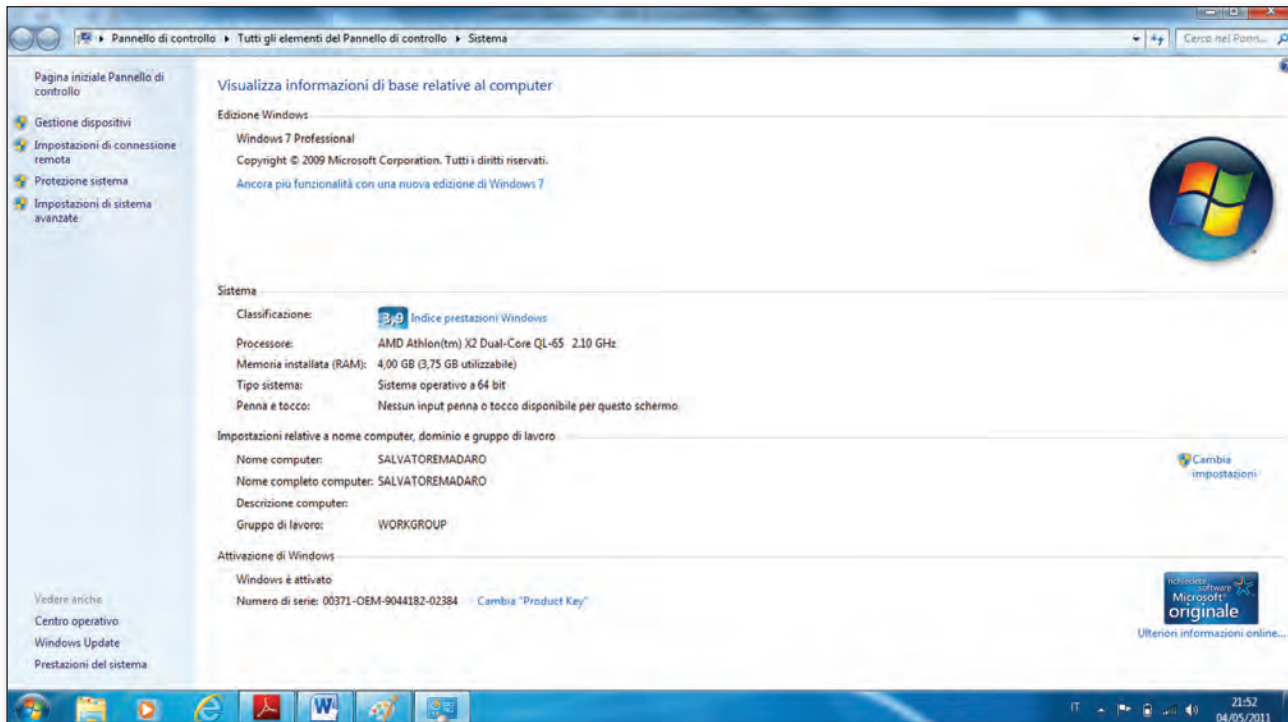
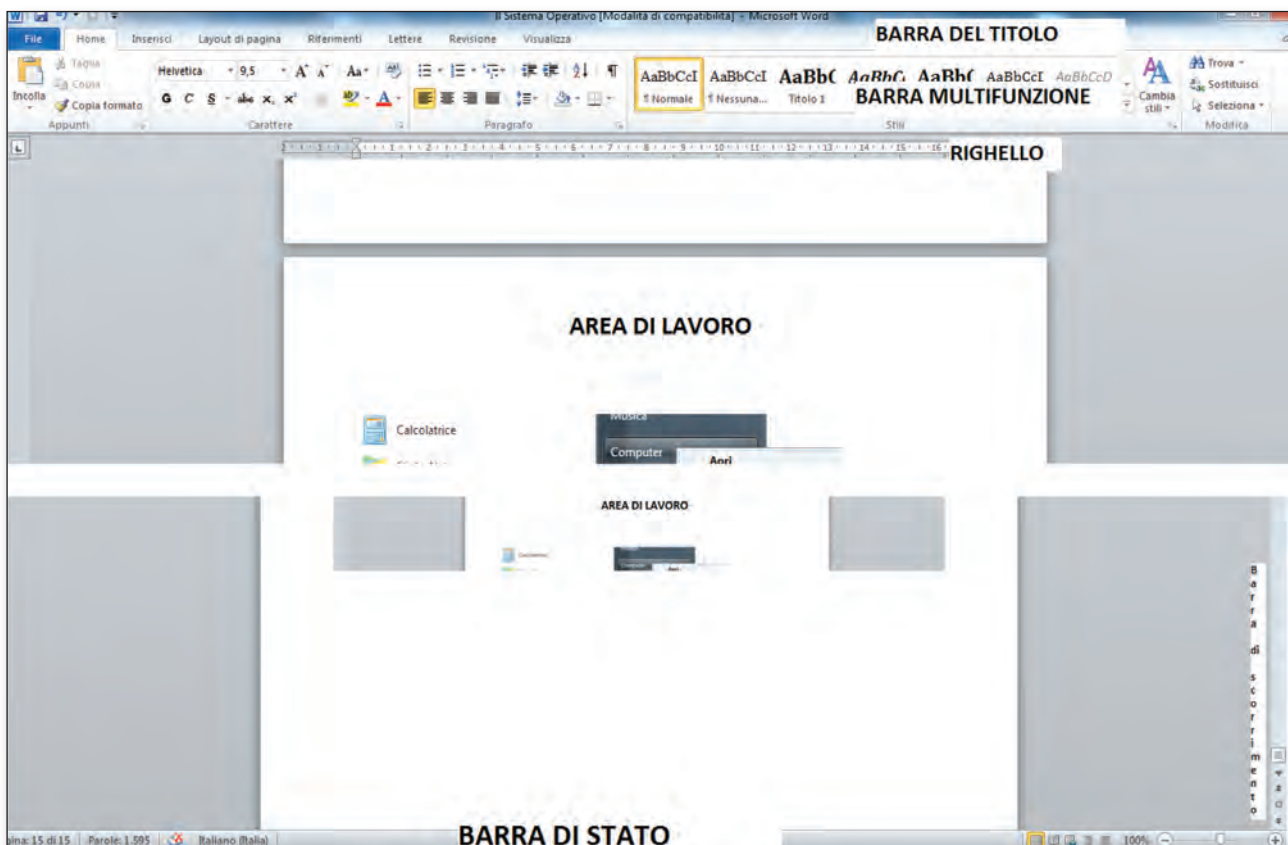
- ***le finestre***
  rappresentazione grafica dei programmi in esecuzione. All'interno di questo elemento grafico si distinguono diversi altri oggetti fra i quali citiamo le *barre*, i *menu*, i *pulsanti*, ecc.;

Barra del titolo
Barra dei menu
Area di lavoro
Barra di stato

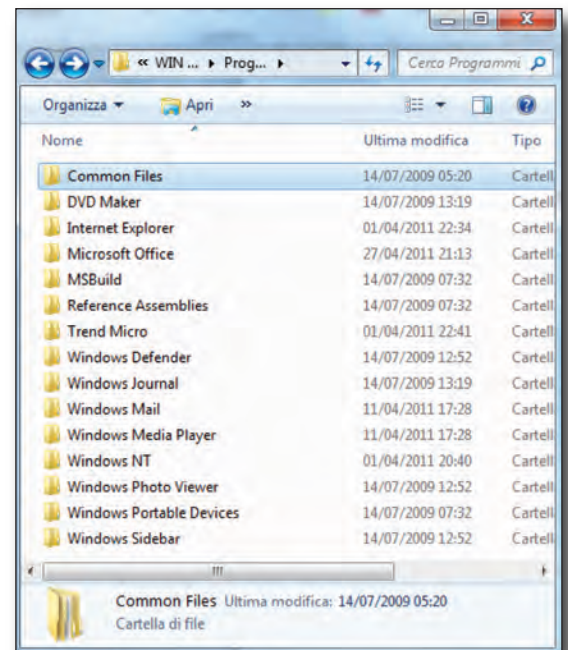
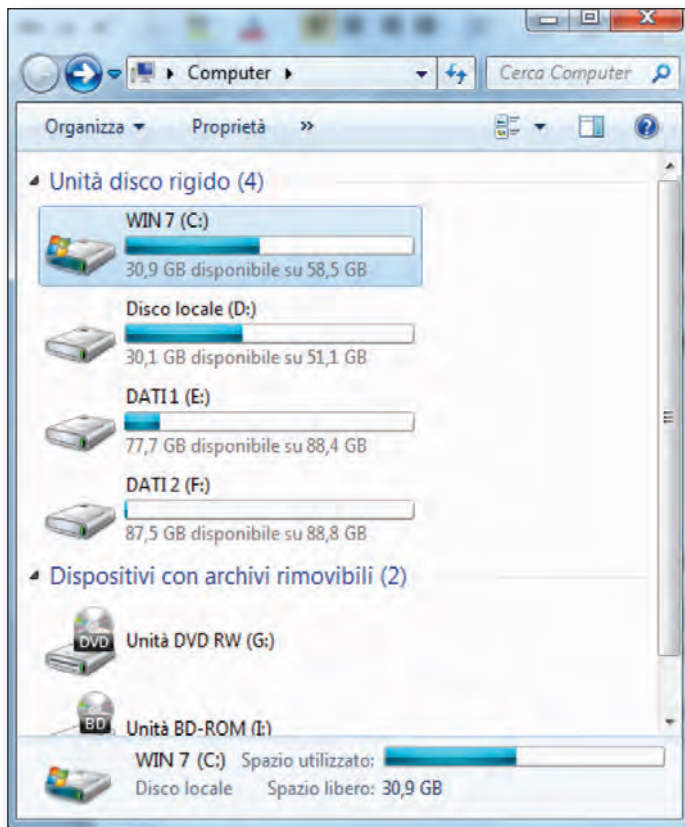
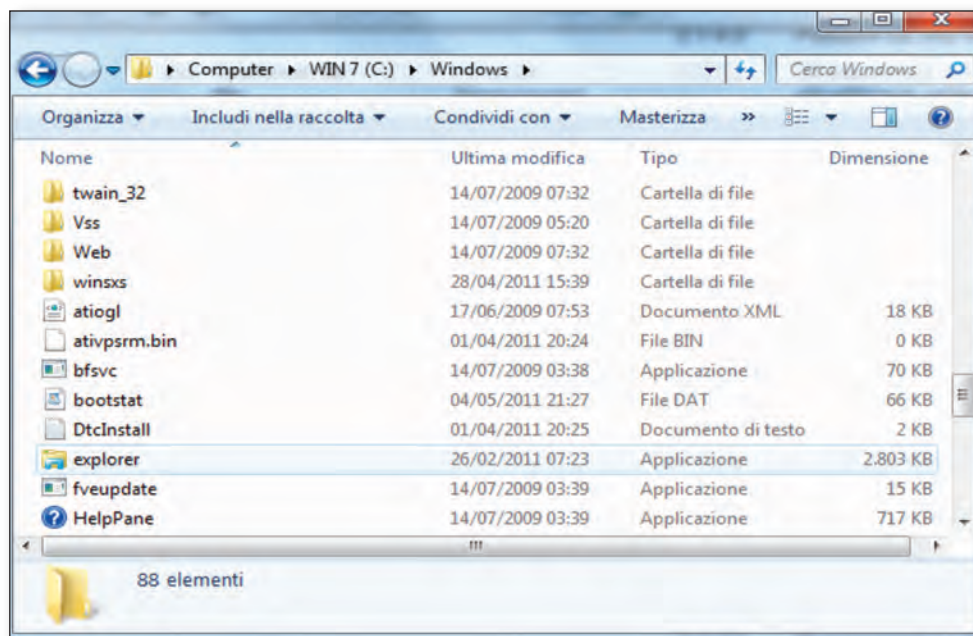
Le principali operazioni del Sistema Operativo mediante l'interfaccia grafica

Mostriamo adesso praticamente come è possibile utilizzare le principali caratteristiche di un sistema operativo; usiamo Windows 7 per presentare le potenzialità di un sistema operativo ad interfaccia grafica perché è tra i più recenti, in rapida espansione tra gli utenti dei personal computer, dei notebook e dei netbook oltre al fatto che è molto stabile e sicuro.

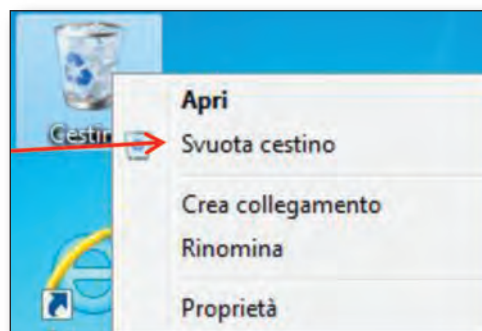
Innanzitutto il **desktop**: è possibile personalizzarlo con un semplice clic destro del mouse; il menù a contestuale ne consente la sua modifica.
Dalla voce "Personalizza" si apre una finestra dalle quale si possono apportare le modifiche desiderate.

Nella **barra delle applicazioni**, in basso a destra, si trovano delle icone che permettono di configurare l'orario, la scheda audio, il collegamento alle reti wireless, la tastiera.

Se invece vogliamo vedere le **caratteristiche tecniche del computer** è sufficiente partire dal pulsante Start → Computer → clic destro sul pulsante → Proprietà

Così è possibile vedere il tipo di processore, la quantità di RAM installata, la versione del Sistema Operativo ecc..

Si vede dal precedente esempio come il sistema operativo opera con le finestre come ogni sistema ad interfaccia grafica; vediamo insieme le parti essenziali di una finestra:

I tre pulsanti in alto a destra di ogni finestra servono rispettivamente nel **ridurre a icona** la finestra nella barra delle applicazione (che si trova in basso al desktop), a **ridimensionare** e a chiudere la finestra.

Se vogliamo vedere la **memoria di massa** disponibile sul PC, le cartelle e i files in esse contenuti è sufficiente seguire il percorso: Pulsante Start → Computer → Doppio clic sul disco del quale interessa vedere il contenuto:

Notiamo nella finestra la presenza di cartelle (directory) e files con la data dell'ultima modifica, la loro dimensione e il loro tipo.

Se tutte queste cartelle e questi files sono importanti allora è necessario pensare a tutelarli mediante delle copie di sicurezza. Questa operazione di copia dei dati ai fini della sicurezza in caso di blocco del computer, di danneggiamento o di furto si chiama **Backup** ed è particolarmente importante quando si tratta di dati di un certo rilievo. I files si possono copiare (selezionare, clic destro, copia, selezione della destinazione, clic destro incolla); si possono spostare (selezionare, clic destro, taglia, selezione della destinazione, clic destro incolla); si possono cancellare (selezionare, clic destro, elimina). Quando si cancella un file o una cartella, questi vengono spostati nel **cestino** dal quale si possono ancora ripristinare in caso di necessità; tale operazione è possibile fintanto che il cestino non viene svuotato.

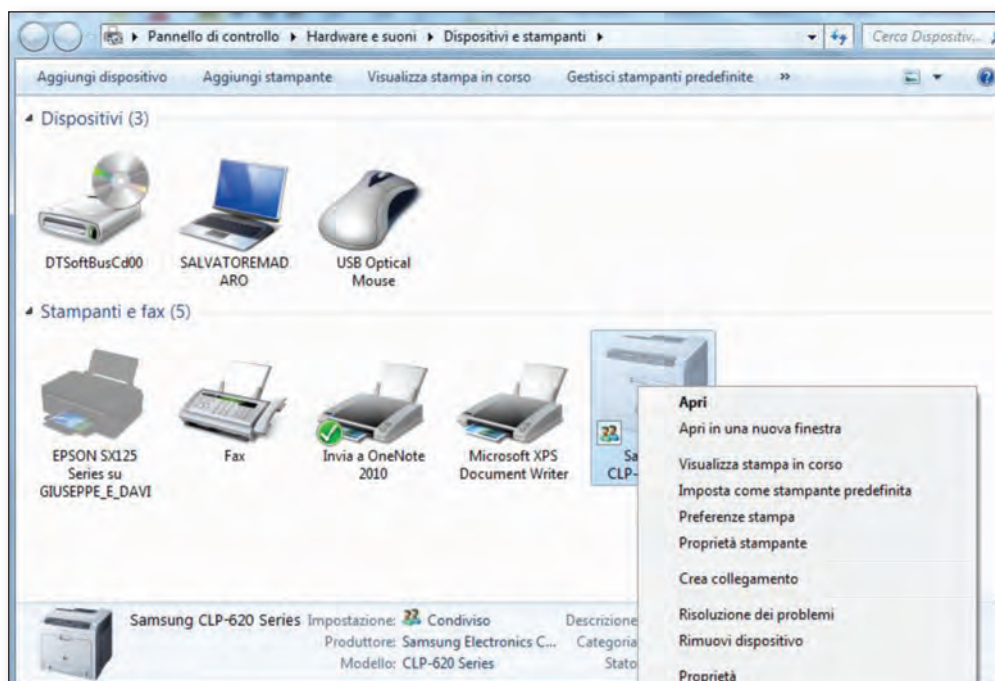
Per ripristinare un elemento dal cestino si procede come segue:
1.doppio clic sull'icona del cestino
2. clic destro sull'icona dell'elemento da vuole recuperare
3. ripristina.

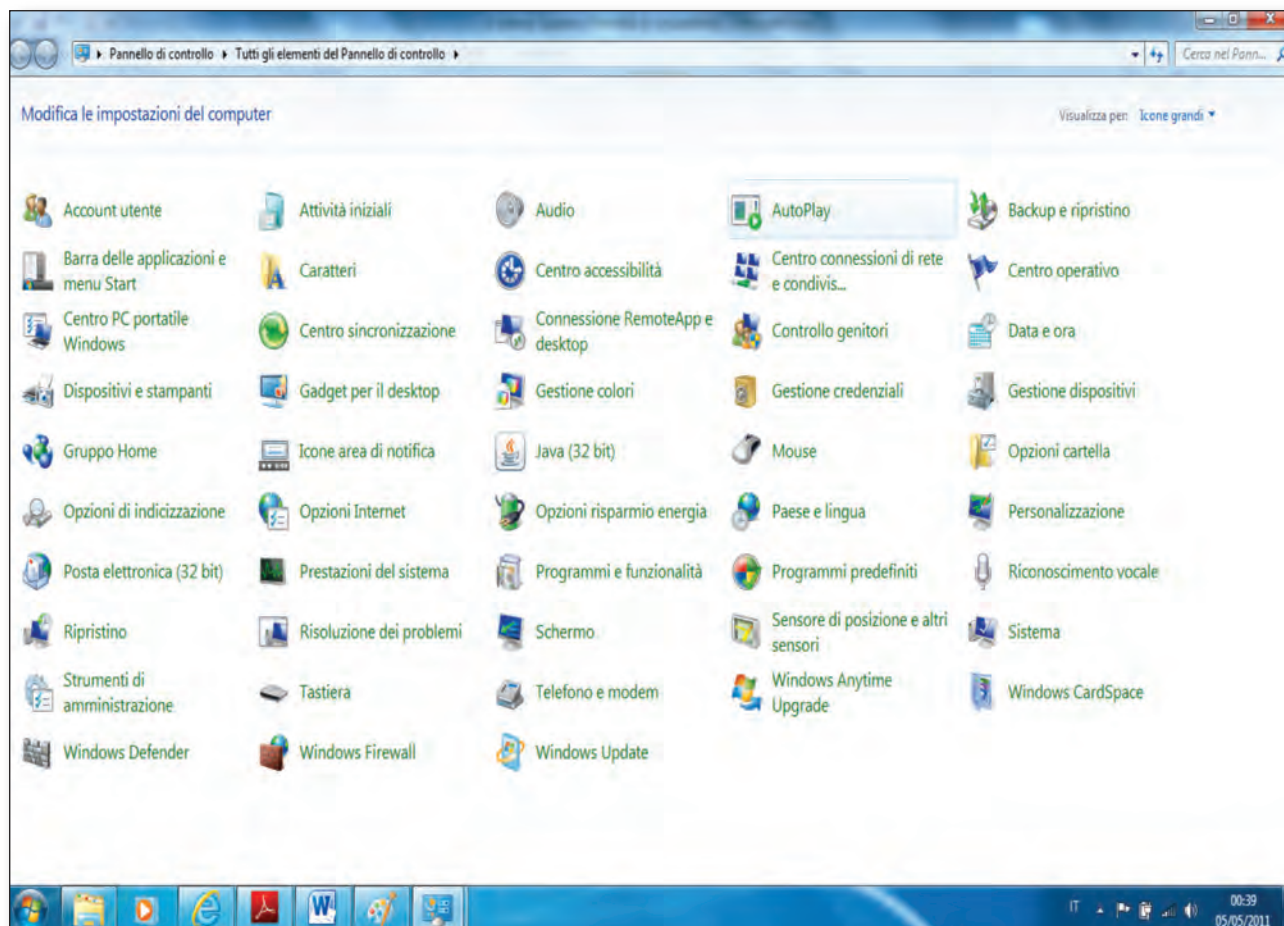
Nei dischi rigidi dei PC sono presenti migliaia di files dispersi all'interno di tantissime cartelle e molte volte è davvero difficile ritrovare qualche documento che avevamo registrato qualche tempo prima; il sistema operativo offre la **funzione "cerca"** che aiuta nella ricerca in diversi modi.

Per attivare questa funzione si parte dal pulsante Start e quindi si inserisce qualche elemento che identifichi il file o la cartella nel campo "Cerca programmi e files".
Quando vogliamo inviare dei files o delle cartelle attraverso Internet oppure vogliamo registrarli su un supporto a bassa capacità, è necessario ridurre il più possibile la loro dimensione; l'operazione che permette di fare questo è la "**compressione**":

clic destro sul file → invia a → cartella compressa.

Quando si copiano o si scaricano files da origini non affidabili è necessario difendersi dai possibili attacchi di **virus informatici** che possono arrecare danni anche gravi ai dati del PC o che possono inviare dei dati sensibili a destinatari sconosciuti che possono usarli a nostra insaputa.

Ci sono degli appositi programmi antivirus che servono a proteggere il nostro computer; alcuni sono "free" (ossia possono essere usati liberamente senza alcun costo) altri a pagamento ma più performanti.

Un elemento molto importante in un sistema operativo è la **gestione delle stampanti**. Nel caso di Windows 7 , per accedere alla relativa scheda bisogna seguire il percorso: pulsante start → dispositivi e stampanti →in questa scheda si può modificare le impostazioni delle stampanti già istallate (con un clic destro sull'icona corrispondente alla stampante prescelta) oppure decidere di istallarne una nuova.

Il **pannello di controllo** (pulsante start → pannello di controllo) è l'interfaccia grafica attraverso la quale è possibile impostare tutte le funzioni del computer; intervenendo in esso è possibile elevare le prestazioni del PC come è possibile attirittura inibire delle funzioni o, addirittura bloccarlo del tutto quindi nel momento in cui si decide di intervenire su di esso bisogna fare molta attenzione.

In questa sede è possibile **gestire gli account** (ossia la possibilità di accesso ai vari utenti del PC anche mediante delle password), la configurazione di accesso alla rete LAN di computer e a Internet e di **rimuovere dei programmi** che riteniamo di non dover più usare.
Questa interfaccia è nello stesso tempo molto intuitiva e potente perché racchiude in se tutte le funzionalità del Sistema Operativo.

5. Come funziona un elaboratore

Autore: **Angelo Oliva**

Competenze	Abilità	Conoscenze
Utilizzare, con autonomia e responsabilità, il personal computer, nelle attività di studio ricerca e approfondimento disciplinare.	*Conoscere le fasi di avviamento del pc e le componenti hardware coinvolte, riconoscere segnalazioni di errori.*	*Architettura e componenti di un computer* *Procedura di avvio, firmware e caratteristiche*

Introduzione

In questo capitolo descriveremo il funzionamento del computer dal suo interno, precisando cosa succede dal momento dell'accensione fino allo stato di pronto, in altre parole quando viene visualizzato il desktop e si attendono i comandi da parte dell'utente. Vedremo quali componenti entrano in gioco senza arrivare a livelli di approfondimento che nelle scuole tecniche vengono affrontati negli anni successivi.
È importante ricordare che la procedura di avviamento del personal computer è assimilabile alla gran parte dei dispositivi digitali oggi in circolazione, come Smartphone, Tablet e SmartTV.

Uno sguardo in Rete:
Su Youtube è pubblicato una videoanimazione che descrive in modo breve ed efficacie la successione degli eventi nell'accensione del computer, realizzata da Marco De Crescenzo e Stefano Di Silvio studenti del Corso di Sistemi Multimediali dell'Università degli Studi di Napoli "Federico II", seguente link: https://youtu.be/2ZlyQAoiAf4

Accensione del computer

Prima di descrivere le fasi di accensione osserviamo la parte hardware coinvolta; all'interno del case in corrispondenza del pulsante Power (figura 1) è presente un fascio di cavi opportunamente cablato (figura 2) che è collegato ad un connettore della scheda madre, nel nostro caso denominato PANEL1 (figura 3).
Normalmente tra i collegamenti (figura 4) troviamo il cavo che consente l'accensione del computer (POWERBTN o POWER SW), quello che consente di alimentare le luci led che segnalano il funzionamento dell'hard disk in lettura/scrittura (HDLED) o la pressione del pulsante RESET, quello che alimenta il piccolo altoparlante interno del case (speaker) e quello che serve alla messa a terra (GND).

Figura 1

Figura 2

Per descrivere meglio gli eventi definiamo un elenco numerato:
1. Power On. Quando premiamo sul pulsante esterno di accensione del pc si susseguono una serie di eventi, il primo consiste nel segnale elettrico che attraverso due fili collegati al pulsante (PWR_BTN e GND) va a cortocircuitare i contatti (pin) corrispondenti al segnale di accensione.

Figura 3

Figura 4

Un tecnico che sta facendo manutenzione ad un computer, potrebbe accendere il pc toccando con la punta di un cacciavite i due pin PWR_BTN del PANEL nello stesso modo che spesso vediamo in alcuni film nelle scene di furti d'auto, in cui il ladro fa toccare i due fili dell'accensione.

Figura 5

Figura 6

2. Si attiva l'alimentore. Il primo dispositivo che si attiva è l'alimentatore (figura 5), che ha un ingresso tramite il quale assorbe corrente alternata dalla rete elettrica (figura 6) che in Europa è di 220V, la trasforma e la stabilizza in corrente continua quindi la distribuisce in uscita per alimentare le diverse componenti hardware del computer, tramite la scheda madre o direttamente attraverso una serie di cavi elettrici che terminano con i vari tipi di connettori a seconda del tipo di componente interna da alimentare, la tensione fornita può andare da 3,3 V a 12 V. L'alimentatore va scelto sommando il totale delle necessità di tensione richiesta da ogni componente, in particolare bisogna fare attenzione se si decide di istallare schede video di fascia alta per il gaming, queste normalmente hanno bisogno di maggiore alimentazione delle schede video standard e quindi bisogna utilizzare un alimentatore adeguato, ricordiamo che anche gli alimentatori sono dotati di ventole di raffreddamento.

3. Scheda Madre e CPU On. L'alimentazione arriva a questo punto a tutti le componenti collegate all'alimentatore, la principale è la scheda madre che è collegata direttamente all'alimentatore da un fascio di 24 cavi che terminano in un connettore detto a ATX, punto di alimentazione principale della scheda madre (figura 7).

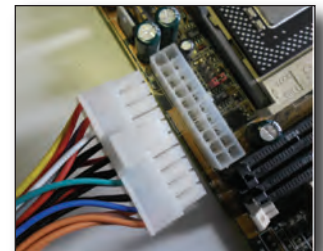



Figura 7

Figura 8

Anche la scheda madre (figura 8) distribuisce la tensione a tutte le componenti ad essa collegate, cioè alle componenti elettriche saldate come condensatori, ai connettori per la CPU, per le Ram e per le schede periferiche aggiuntive come la Scheda Video o la Scheda di Rete. La rete attraverso la quale giunge la tensione a tutte le componenti presenti sulla scheda è ben visibile nella figura 9.

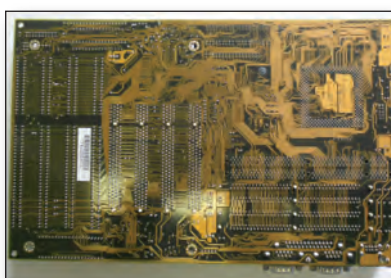



Figura 9

Tra queste componenti si attiva la CPU, Central Processing Unit (figura 10), il microprocessore più importante di tutto il sistema, costruito con un sofisticatissimo processo industriale di miniaturizzazione dei circuiti, oggi in commercio esistono processori che contengono anche più di un miliardo di transitor ed eseguono miliardi di operazioni elementari al secondo, tale parametro si misura in Ghz, 1 Ghz corrisponde ad un miliardo di operazioni al secondo. Durante il funzionamento nel microprocessore possono svilupparsi elevate temperature interne, fenomeno legato alla resistenza nel passaggio di corrente elettrica attraverso i conduttori; tale problema viene risolto attraverso l'uso di dissipatori di calore attivi, cioè ventole di raffreddamento, o di dissipatori passivi cioè griglie simili a radiatori che favoriscono la dispersione del calore in modo statico o spesso dalla combinazione di entrambe le tipologie, la ventola viene montata insieme al dissipatore passivo (figura 12-13) e attraverso la sua rotazione aspira aria calda. I dissipatori passivi sono di solito incollati sui microprocessori con un particolare prodotto chimico, la pasta termoconduttiva (figura 11) che permette il passaggio di calore, al contrario delle normali colle che tendono a isolare e sigillare.

Figura 10

Figura 11

Figura 12

Figura 13

4. Fase di POST. la CPU, una volta attivata esegue il BIOS (Basic Input Output System), un programma di inizializzazione installato nel computer dall'azienda produttrice, quello che viene chiamato generalmente firmware. Il firmware esiste in ogni dispositivo digitale, dallo smartphone alla tv non necessariamente smart, dalla lavatrice al frigorifero; ogni dispositivo digitale all'accensione prevede l'esecuzione di una procedura di avviamento, che nel computer è definita in questo software, personalizzabile in base alle esigenze dell'utente semplicemente accedendo al SETUP del programma attraverso la digitazione di uno o una combinazione di tasti durante la sua esecuzione, per esempio il tasto Canc o Del (Figura 14).

Figura 14

Figura 15

Inizialmente questo software era memorizzato su ROM (Read Only Memory), una memoria a sola lettura, pertanto non poteva essere aggiornato, neanche nel caso in cui venivano scoperti malfunzionamenti detti " bug" nel programma, l'unica soluzione era sostituire tutto il chip della ROM. Una serie di progressi sulle tecnologie di produzione di tali memorie ha portato oggi all'adozione delle EEPROM (Eletrical Erasable Programmable ROM – figura 15), memorie aggiornabili con una particolare procedura, che dall'inglese "to Flash a Rom" viene tradotta in Flashare la Rom.

Adesso è quindi possibile aggiornare il Bios, con nuove versioni corrette e ottimizzate, quest'operazione non può essere considerata di ordinaria manutenzione ma deve essere fatta solo quando strettamente necessario, di solito quando consigliato dall'azienda produttrice e in condizioni di sicurezza, per esempio utilizzando gruppi di continuità perché se dovesse verificarsi un problema come la mancanza di corrente durante l'operazione, non si potrebbe portare a termine la procedura e sarebbe necessario sostituire la EEPROM con un Bios funzionante a bordo. Le EEPROM oggi si trovano in tantissimi dispositivi e sistemi, per esempio nelle centraline elettroniche delle automobili che gestiscono numerose funzioni come l'iniezione elettronica e il sistema di controllo della trazione, molte officine si sono attrezzate per effettuare il cosiddetto Tuning Meccanico che consiste nel riprogrammare il software presente nella eeprom aumentando per esempio la potenza del motore a basso regime di giri.

Il BIOS prevede una serie di controlli di funzionamento di dispositivi indispensabili e la presenza di dispositivi non indispensabili connessi, per esempio viene rilevata la presenza e la dimensione della RAM (memoria centrale) installata sulla scheda madre, la presenza e la dimensione dell'hard disk principale (primary master) l'eventuale presenza di ulteriori hard disk o unità ottiche come lettori o masterizzatori CD DVD. Questa serie di controlli è chiamata fase di POST Power On Self Test, tali test vengono visualizzati sul monitor (figura 16) a meno che tale funzione non sia stata disabilitata da BIOS. In caso di problemi che non consentono l'avviamento il programma attiva l'emissione di una serie di beep, dal cui numero è possibile interpretare il tipo di errore, ogni casa produttrice di BIOS usa differenti codici sonori per indicare i diversi tipi di problemi hardware, per conoscere la corrispondenza tra codici sonori e malfunzionamenti hardware bisogna consultare la documentazione della scheda madre, per esempio nell'Ami Bios due beep corrispondono a errore della Ram.

Figura 16

Puoi simulare un errore nella fase di POST disconnettendola tastiera dal personal computer, all'accensione dovresti sentire il numero di beep corrispondenti al messaggio keyboard error.

Se tutti i controlli vanno a buon fine il BIOS piloterà l'emissione di un solo beep e concluderà la fase di POST richiedendo il caricamento del sistema operativo da una unità di memoria di massa, in genere l'hard disk "C:", anche la sequenza ordinata di unità su cui cercare il sistema operativo detta Sequenza di Boot è impostata nel BIOS e può essere modificata secondo le proprie esigenze.

5. Fase di Bootstrap. Il sistema operativo, o meglio ancora la parte indispensabile per la gestione del computer chiamata nucleo o "kernel" viene caricato nella memoria centrale RAM, questa viene chiamata fase di Bootstrap o semplicemente Boot, al termine il nostro PC visualizzerà il desktop cioè l'interfaccia grafica che consente attraverso dispositivi di input come mouse, tastiera, touchpad o touchscreen l'immissione di comandi da parte dell'utente.

Programmi e Processi

Terminata la fase di Bootstrap, il nostro computer è pronto per ricevere comandi dall'utente come ad esempio visualizzare il contenuto di una unità di memoria o di una cartella, oppure richiedere l'esecuzione di un'applicazione, spesso in questo contesto il computer veniva chiamato macchina virtuale intendendo con questo termine il computer nella sua funzione, cioè non vedendolo più come un sistema complesso ma come un unicum che è dedicato ad una particolare funzione, per esempio il computer utilizzato in uno studio di architettura solo con un software CAD per la progettazione, nel momento in cui è acceso con tale programma in esecuzione viene inteso come la macchina (virtuale) che consente la realizzazione di progetti.

Oggi il significato di macchina virtuale (VM) è parzialmente cambiato, perché le potenze raggiunte dai computer consentono di emulare con il software appropriato altre macchine fisiche, per esempio su una macchina con un determinato sistema operativo si possono emulare i comportamenti di macchine fisiche diverse con sistemi operativi diversi, oggi è molto diffuso emulare Tablet e Smartphone con sistema operativo Android su computer basati su altri sistemi operativi, permettendo di utilizzare su computer applicazioni mobile.
Come già detto nel capitolo 1 i computer sono basati sul **Modello di Von Neumann** nella quale è presente la memoria centrale che ha la funzione di contenere **programmi in esecuzione** e i relativi **dati** e **l'Unità Centrale di Elaborazione**, la CPU che ha il compito di elaborare una alla volta le istruzioni prelevate dalla memoria centrale attraverso delle linee di comunicazione dette BUS secondo uno schema ciclico detto "ciclo d'istruzione" composto da due fasi:

1. La Fase di fetch (estrazione): consiste nell'estrarre l'istruzione dalla RAM (memoria centrale) e trasferirla attraverso linee di comunicazione (bus) nella CPU (Central processing unit)
2. La Fase di Execute: La CPU o microprocessore è l'unica unità in grado di eseguire istruzioni, l'esecuzione può richiedere diversi operazioni elementari come l'estrazione dei dati e l'esecuzione di calcoli logico aritmetici, al termine della fase di esecuzione si continuerà con l'estrazione della prossima istruzione fino a quando l'esecuzione del programma non termina con l'istruzione End.

La macchina Von Neumann

INTERFACCIA I/O
INTERFACCIA I/O
INTERFACCIA I/O
BUS
CPU
MEMORIA

Gli elementi funzionali della Macchina sono quattro:

- ✓ **Unità di elaborazione (CPU, Central Processor** Unit) dove vengono eseguite le istruzioni una per volta dei processi in esecuzione
- ✓ **Memoria centrale** dove vengono caricati dati iniziali, temporanei e programmi durante il funzionamento,
- ✓ **Periferiche** le unità che consentono di immettere dati nel sistema e di ricevere messaggi e risultati in uscita
- ✓ **Bus di sistema** il canale che collega tutti i componenti del sistema, permettendo loro di dialogare

Quando clicchiamo su un'icona che corrisponde alla partenza di un programma, per esempio word, paint, un gioco o altro, il sistema operativo interpreta questo click, caricando una copia del programma che si trova su una memoria di massa come l'hard disk, nella memoria centrale RAM, da questo momento in poi non si parla più di programma ma di processo, **il processo corrisponde all'immagine del programma in esecuzione**; a computer acceso nella memoria RAM sono presenti, oltre al nucleo del Sistema Operativo, tutti i processi in esecuzione; prima dell'avvento dei processori con più Core, i computer potevano eseguire un'unica istruzione per volta proprio perché l'unità capace di eseguire le istruzioni era una sola, la sensazione di avere più processi eseguiti contemporaneamente è dovuta al fatto che la risorsa CPU veniva condivisa nel tempo dai processi, piccole quantità di tempo CPU erano dedicate a ciascun processo sequenzialmente ma la velocità d'esecuzione era tale da dare la sensazione di un'esecuzione parallela, grazie alla velocità del microprocessore che oggi varia da 1 a oltre 3 Gigahertz, (ad 1 Ghz corrispondono un miliardo di operazioni elementari al secondo). Il progresso tecnologico ha portato alla produzione di processori multicore, cioè all'integrazione di più processori nello stesso chip, un processore quadcore può essere considerato come se fosse costituito da quattro processori separati, quindi è possibile che esegua quattro processi diversi contemporaneamente cioè è possibile che realizzi **l'esecuzione parallela**.

6. Algoritmi e Diagrammi di Flusso

Autore: **Angelo Oliva**

Competenze	Abilità	Conoscenze
Utilizzare, con autonomia metodologica ed esecutiva, procedure e tecniche per trovare soluzioni efficaci ed efficienti, in relazione a semplici problemi nei campi di propria competenza.	*Rappresentare la soluzione di un problema con diagrammi di flusso.* *Analizzare e risolvere semplici problemi con i principi della programmazione strutturata, problem solving e astrazione.*	*Concetto di variabile e di algoritmo.* *Fasi risolutive di un problema e rappresentazione con diagrammi di flusso o pseudocodice.* *Fondamenti di programmazione strutturata.*

Introduzione

Francobollo commemorativo del 1200° anniversario della nascita matematico persiano, stampato in Unione Sovietica il 6 settembre 1983

Non li vediamo, ma sono sempre all'opera e sono diventati fondamentali nella nostra società, se prendiamo un volo o cerchiamo l'anima gemella, se facciamo acquisti on line o utilizziamo i social network, se prenotiamo una visita medica o un farmaco, anche se semplicemente accendiamo lo smartphone ci sono sempre degli **algoritmi** che operano con discrezione "dietro le quinte" e che hanno acquistato un ruolo chiave nella vita quotidiana, ma essendo invisibili spesso non li consideriamo e non ne consideriamo l'importanza.

Di certo computer, smartphone e internet sono il regno degli algoritmi, ma non dobbiamo associarli solo ad essi, anzi, essendo essenzialmente una serie di istruzioni progettate per risolvere un problema, si può affermare che gli algoritmi esistono da prima della nascita di queste tecnologie; La natura stessa opera moltissime volte in modo algoritmico, basti pensare ai metodi di costruzioni di nidi e tane, alla tela del ragno, o alle tecniche di caccia per riconoscere precise sequenze di azioni che regolano questi e tanti altri fenomeni;

Il termine algoritmo deriva dalla trascrizione latina del nome del matematico persiano vissuto nel IX secolo D.C. **Al-Khuwarizmi**, considerato uno dei primi autori che abbia fatto riferimento al concetto di risoluzione di problemi in un numero finito di passi nel suo libro "Regole di ripristino e riduzione". Lo stesso matematico è ricordato anche per aver scritto un altro libro dal cui titolo deriva il termine algebra.

Gli algoritmi formulati dall'uomo risalgono a moltissimi anni fa, il ritrovamento del papiro di Ahmes, noto anche come papiro di Rhind, l'egittologo scozzese che lo acquistò a Luxor nel 1858, fa datare i primi algoritmi al 2850 A.C. circa, Ahmes era uno scriba, autore di questo papiro che chiamò "Regole per ottenere la conoscenza di tutte le cose oscure", nel quale sono contenuti una serie di problemi e soluzioni tra cui il famoso problema n. 79 noto come problema dei sette gatti e l'ancor più interessante metodo di risoluzione che gli egizi usavano per eseguire la moltiplicazione. Il papiro di Ahmes risale al 1650 a.c. circa ma egli stesso dichiara di aver riportato contenuti di altri papiri di circa 12 secoli prima.

intuito la necessità di un sistema per istruire la macchina attraverso una serie di comandi e per dimostrare come doveva funzionare definì una serie di istruzioni per generare una nota serie numerica, i numeri di Bernoulli.

Ada Lovelace ritratto 1840

L'algoritmo più noto tra quelli più antichi è sicuramente quello del **Massimo Comun Divisore** (circa 300 A.C.), redatto in forma scritta dal matematico greco **Euclide** (problema del MCD in versione geometrica) mentre il primo algoritmo progettato per essere potenzialmente eseguito da una macchina è attribuito ad Augusta Ada King, Contessa di Lovelace, meglio nota come **Ada Lovelace**; nata a Londra nel 1815, matematica inglese, incontrò Charles Babbage mentre progettava la sua Macchina Analitica che però non riuscì mai a terminare; Babbage chiese ad Ada Lovelace di tradurre un articolo sul suo progetto pubblicato in francese nel 1842 dallo scienziato piemontese Luigi Malerba, la Lovelace non solo lo tradusse ma ne aggiunse delle note di suo pugno, sorprendenti perché dimostrano che aveva intuito la necessità di un sistema per istruire la macchina attraverso una serie di comandi e per dimostrare come doveva funzionare definì una serie di istruzioni per generare una nota serie numerica, i numeri di Bernoulli.

Un altro famoso algoritmo progettato ma non eseguito su una macchina è quello relativo al gioco degli scacchi, ad opera di **Alan Turing**, matematico inglese precursore delle moderne ricerche sull'intelligenza artificiale e noto soprattutto per aver lavorato ad una macchina elettromeccanica in grado di violare i messaggi criptati generati dalla macchina Enigma utilizzata dai tedeschi durante la seconda guerra mondiale; Turing, non avendo a disposizione una macchina in grado di eseguire le istruzioni del suo algoritmo sugli scacchi eseguì egli stesso le istruzioni in una storica partita giocata a Manchester nel 1952 contro Glennie Alick, un suo collega informatico, la partita durò diverse settimane perché Turing doveva eseguire i calcoli dell'algoritmo con carta e penna, alla fine Glennie ne uscì vincitore. È interessante ricordare anche la prima volta che un computer ed il suo programma hanno battuto un campione del mondo di scacchi, è stato il 10 Febbraio 1996, il russo Garry Kasparov affrontò il Computer **"Deep Blue"** appositamente progettato da IBM; oggi la potenza di calcolo dei computer e gli algoritmi progettati sono così potenti che la sfida non è più uomo macchina, ma macchina macchina (Top chess Engine Championship o World Computer Chess Championship).

Deep Blue, IBM.
Autore: James the photographer [CC BY 2.0 (https://creativecommons.org/licenses/by/2.0)], via Wikimedia Commons

Problemi e soluzioni

Vi sarà capitato di utilizzare una "macchinetta" che distribuisce merendine e bibite, per poter prendere un prodotto dobbiamo eseguire una **precisa sequenza di istruzioni**, **ordinata**, nel senso che la selezione del prodotto deve essere fatta dopo altre istruzioni quali l'introduzione delle monete, **finita** nel senso che terminerà comunque in un tempo misurabile, dando un risultato che dipende dalle operazioni fatte dal "cliente" e dalla presenza o meno del prodotto richiesto. Le operazioni del cliente, che possiamo definire ingressi o input, cioè l'introduzione del denaro e la pressione dei pulsanti corrispondenti al codice del prodotto desiderato, vengono "elaborate" producendo un risultato; la macchinetta, eseguendo un processo che possiamo definire algoritmo, emetterà delle uscite ben precise e **determinate**, che corrisponderanno all'emissione del prodotto richiesto, oppure ad un messaggio nel caso in cui il prodotto è esaurito o ad un altro messaggio ancora se l'importo inserito è insufficiente, consentendo in questi due ultimi casi di procedere con altri ingressi che possono essere la selezione di un altro prodotto, la restituzione dell'importo inserito o l'integrazione dell'importo insufficiente insieme con la selezione del prodotto.

Il processo, cioè la sequenza di operazioni che la macchina esegue sono definite da un algoritmo. L'algoritmo esprime le ***azioni*** da svolgere su determinati ***oggetti*** al fine di produrre gli ***effetti*** attesi, un'azione che produce un determinato effetto è chiamata **istruzione** e gli oggetti su cui agiscono le istruzioni sono i **dati**, cioè l'insieme delle informazioni che devono essere elaborate, secondo le modalità descritte dalle istruzioni, per produrre altri dati. Potremmo dire cioè che l'algoritmo consiste nella trasformazione dei dati di un insieme d'ingresso (**dati di input**) in dati di un insieme di uscita (**dati di output**).

I dati possono essere distinti in **costanti** (valori che restano sempre uguali nelle diverse esecuzioni dell'algoritmo) e **variabili** (contenitori di valori che variano ad ogni esecuzione dell'algoritmo), definizioni che approfondiremo più avanti.

Più volte abbiamo detto che l'algoritmo deve risolvere un problema generale o una classe di problemi, è bene chiarire cosa s'intende per problema. Un problema nasce generalmente da un bisogno, per esempio se volessi preparare una torta speciale per il mio compleanno e non sono capace mi dovrò rivolgere ad un esperto, potrei dire che io in qualità di **cliente**, ho un **problema**, cioè voglio preparare una particolare torta, mi hanno suggerito di rivolgermi ad un esperto pasticciere, colui che risolverà il mio problema e che possiamo chiamare appunto **risolutore**; Il risolutore per risolvere il problema, analizza la situazione, per esempio chiedendo quale tipo di torta desidero, per quante persone, se la cucina è attrezzata di tutti gli strumenti necessari per la preparazione, e così via; al termine di questa **analisi** se ha avuto tutte le risposte necessarie potrà definire la **ricetta** cioè quali sono le istruzioni da eseguire per preparare la torta, in altre parole **l'algoritmo**; a questo punto una qualsiasi persona esperta in cucina, che possiamo chiamare **esecutore** può eseguire le istruzioni della ricetta ottenendo come risultato finale la torta desiderata. È interessante notare che se volessi di nuovo la stessa torta, ho già la ricetta, cioè l'algoritmo, se sono in grado di cucinare potrò ripetere la preparazione tutte le volte che voglio, così come potrebbe farlo qualsiasi esecutore. Facciamo ancora un esempio, supponiamo di dover tinteggiare le pareti di una stanza (**problema**), mi reco in un negozio che vende vernici l'addetto alla vendita (**risolutore**), mi spiega come fare dopo avermi posto una serie di domande (**analisi**), mi chiede se ho gli attrezzi giusti, quale tipo di vernice preferisco e soprattutto mi spiega che ogni confezione consente di verniciare una determinata superficie, pertanto sapendo le misure della stanza potrò decidere quante confezioni comprare, a questo punto io stesso potrei procedere alla verniciatura (**esecutore**) seguendo le istruzioni su come preparare la vernice, come usare i pennelli e così via; anche in questo caso se dovessi tinteggiare un'altra stanza basta sapere la superficie, perché ho "l'algoritmo" che mi permette di calcolare quanta vernice comprare.

Un problema è ben posto se:
1. è chiaro l'obiettivo da raggiungere
2. i dati di partenza sono noti e sufficienti
3. il problema sia risolubile da parte chi lo affronta

Nel linguaggio comune il termine algoritmo e il termine programma sono quasi sempre utilizzati come se fossero sinonimi, per fare chiarezza il programma ha come esecutore un computer ed è costituito da una serie di istruzioni scritte rispettando regole rigorose di sintassi, affinché l'elaboratore possa eseguirle correttamente, si tratta del linguaggio macchina. Partendo dal problema è molto complesso scrivere direttamente le istruzione del programma che lo risolvono, passo essenziale per arrivare al programma è progettare l'algoritmo risolutivo del problema, successivamente sarà molto più semplice realizzare il relativo programma attraverso la traduzione o codifica che dir si voglia delle istruzioni dell'algoritmo in istruzioni di un linguaggio di programmazione, che poi l'elaboratore stesso tramite un altro programma trasformerà in sequenze di istruzioni in linguaggio macchina, cioè nel programma eseguibile dal computer.

Definizioni e proprietà degli Algoritmi

A questo punto possiamo rispondere alla domanda "Cos'è un Algoritmo? ":

> L'algoritmo è **"...un insieme di istruzioni che definiscono una sequenza di operazioni mediante le quali si risolvono tutti i problemi di una determinata classe".**

E poi "quali sono le proprietà caratteristiche che un algoritmo deve possedere? ":

1. **Generalità**

Non necessariamente un insieme di istruzioni è un algoritmo, supponiamo di voler sapere l'area di un triangolo di base 5 e altezza 10, se definisco come istruzione di risoluzione

Area = 5 x 10;

l'esecutore potrà risolvere solo un particolare problema cioè l'area del triangolo di quelle precise misure, base 5 e altezza 10 e non una classe di problemi, cioè l'area di tutti i triangoli, che risolverei se dopo aver chiesto i valori della base e dell'altezza utilizzassi come istruzione

Area = base x altezza;

questa è uno dei requisiti di una serie di istruzioni per poter essere considerata un algoritmo, è la proprietà di generalità, un algoritmo deve risolvere una famiglia o classe di problemi, solo in questo caso risulta utile e riutilizzabile.

2. **Finitezza e Terminazione**

In primo luogo si chiede che il **numero di istruzioni** che compongo l'algoritmo sia **finito**, un algoritmo non può contenere un numero infinito di istruzioni e del resto non sarebbe nemmeno rappresentabile. La finitezza descrive anche un'altra situazione, a volte chiamata **Terminazione**, ed è la seguente: supponiamo di definire una sequenza di istruzioni che chiede, dato come valore d'ingresso diverso da zero di calcolare il suo doppio e ripetere tale operazione sul risultato fino a quando non otteniamo un numero dispari; noto che il doppio di qualsiasi numero diverso da zero è sempre pari l'esecuzione continuerà all'infinito, questa procedura non può essere considerata un algoritmo perché la finitezza richiede che **l'esecuzione della procedura deve terminare in un tempo finito per ogni possibile insieme di valori d'ingresso**.

3. **Determinismo**

In una ricetta spesso troviamo istruzioni del tipo "aggiungere sale quanto basta", questa è un istruzione non prevista in un algoritmo perché ambigua, nel senso che il "quanto basta" è una quantità indeterminata, soggettiva e a causa di tali istruzioni il risultato della ricetta può variare da esecutore a esecutore; le istruzioni presenti in un algoritmo devono essere definite senza ambiguità. **Un algoritmo** eseguito più volte e da diversi esecutori, **con gli stessi valori d'ingresso deve giungere a medesimi risultati, indipendentemente dall'esecutore o dall'istante in cui viene eseguito**.

4. **Realizzabilità pratica**

L'esecutore deve essere in grado di eseguire l'algoritmo con le risorse a sua disposizione, cioè deve disporre non solo delle informazioni, ma anche della tecnologia e delle abilità. Se descrivo l'algoritmo che nell'ambito del disegno tecnico mi consente di costruire un esagono, per poterlo eseguire devo disporre di tutti gli strumenti previsti, per esempio del compasso, e devo saperlo usare. Se si tratta di un algoritmo che deve essere eseguibile da una macchina, non vi devono essere comandi che la macchina non riconosce.

Una definizione più formale dell'algoritmo è la seguente:

L'algoritmo è un insieme ben ordinato di operazioni non ambigue ed effettivamente calcolabili, progettato per risolvere una determinata classe di problemi che, quando eseguito, produce un risultato e termina in una quantità finita di tempo."

dove:

insieme ben ordinato significa che l'ordinamento delle operazioni da eseguire deve essere chiaro e non ambiguo, l'esecuzione deve procedere senza ambiguità da una operazione alla successiva.

operazioni non ambigue calcolabili significa che tutti i passi devono essere chiari ed eseguibili per chi esegue le operazioni.

produce un risultato, gli algoritmi risolvono problemi, se non fosse previsto un risultato di uscita non avrebbe senso progettare un algoritmo, una delle classiche domande nell'informatica è "cosa fa l'algoritmo?"

Termina in una quantità finita di tempo come già detto, l'algoritmo deve produrre un risultato dopo l'esecuzione di un numero finito di operazioni.

Gli algoritmi non sono che formule da eseguire passo dopo passo, per progettarli serve creatività e intelligenza, per usarli basta eseguire delle istruzioni, per questo si adattano perfettamente ai computer che altro non sono che macchine che eseguono grandi quantità di istruzioni ad altissima velocità. Per somministrare **un algoritmo** ad un computer, questo **deve essere tradotto nel linguaggio eseguibile dalla macchina**, bisogna cioè codificare l'algoritmo in un programma scritto in linguaggio di programmazione che sarà poi tradotto dallo stesso computer in una serie di istruzioni in formato binario, cioè sequenze di bit, che potranno essere eseguite ogni volta che vogliamo. Risulta spesso difficile passare dall'analisi del problema direttamente alla sua soluzione scritta in un linguaggio di programmazione, è sempre consigliabile prima definire l'algoritmo risolutivo e poi codificarlo con un linguaggio di programmazione adatto al tipo di problema, vediamo quindi come si possono formalizzare e rappresentare gli algoritmi.

Linguaggi di descrizione degli Algoritmi

Se leggiamo una ricetta possiamo individuare le operazioni da svolgere, ma a volte la sequenza non è chiara, spesso dobbiamo eseguire più operazioni contemporaneamente, per esempio controllare una cottura nel forno e preparare la crema, inoltre troviamo anche istruzioni ambigue.

Per poter descrivere un algoritmo, in modo che possa essere comunicato ad altri esecutori, o che possa essere "ricordato" per poterlo eseguire a distanza di tempo, esistono diversi strumenti chiamati **Linguaggi di descrizione degli algoritmi**. Non possiamo descrivere gli algoritmi con il nostro linguaggio normale, il linguaggio naturale, perché pur essendo di facile interpretazione per l'uomo, spesso presenta delle caratteristiche di ambiguità di cui non ci rendiamo conto. Per esempio se ci dicono "vai al supermercato e compra due confezioni di latte, se ci sono le uova prendine sei", noi acquisteremmo due confezioni di latte o due confezioni di latte e sei uova; un esecutore che non interpreta il linguaggio naturale invece acquisterebbe o due o sei confezioni di latte, condizionando la scelta alla presenza delle uova, l'istruzione in linguaggio naturale è ambigua.

I linguaggi di descrizione degli algoritmi devono consentire nel loro utilizzo il rispetto di tutte le proprietà descritte degli algoritmi ed in particolare devono rappresentare il **Flusso di Controllo** delle operazioni consentendo di stabilire senza ambiguità qual è la prossima operazione da eseguire, ciò è possibile grazie all'uso di un **Lessico** (vocabolario) limitato per definire dati e comandi, di una rigida e ben definita **Sintassi**, cioè l'insieme di regole di composizione per la scrittura delle istruzioni e di una **Semantica** cioè l'insieme di regole per l'interpretazione corretta delle istruzioni;

Possiamo classificare i linguaggi di descrizione degli algoritmi in due tipi:

1. **Grafici**, che fanno uso di figure geometriche frecce oltre che di parole, sono quelli che si prestano facilmente ad una comprensione dell'algoritmo e al suo flusso di esecuzione, il più noto linguaggio grafico è il Diagramma di Flusso (Flow chart), spesso chiamato schema a blocchi.

2. **Lineari**, che descrivono l'algoritmo esclusivamente in modo testuale cioè a parole, sono più compatti di quello grafici ma non consentono la stessa velocità di comprensione, usano quindi un Metalinguaggio o Pseudo-linguaggio, chiamato spesso Pseudo-codifica.

I Diagrammi di Flusso

I **diagrammi di flusso** o flow-chart permettono di descrivere gli algoritmi attraverso una serie di simboli grafici, chiamati blocchi logici, che si differenziano per la loro forma in base alla funzione che svolgono, questi blocchi sono uniti da archi orientati (frecce) che indicano la direzione del flusso di elaborazione, definendo cioè in modo non ambiguo l'ordine di esecuzione delle istruzioni. I diagrammi di flusso rientrano nella categoria dei linguaggi formali, che adottano come **Lessico** i seguenti simboli grafici:

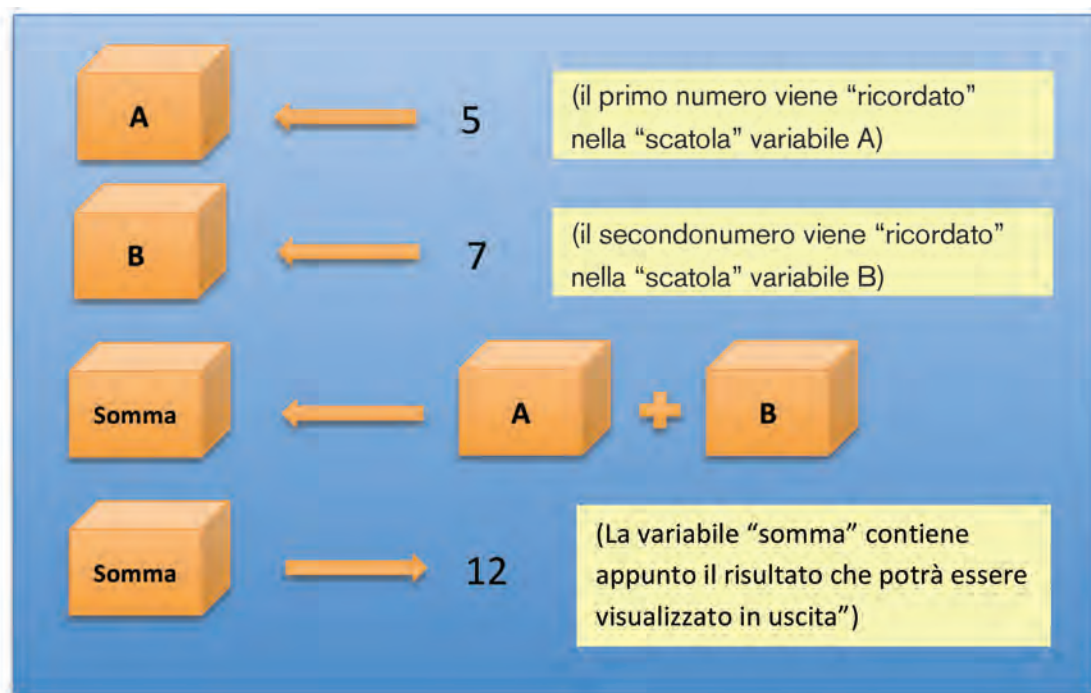
• *Ellissi: indicano l'inizio e la fine dell'algoritmo*	Inizio Fine
• *Parallelogramma: indica un operazione di input output, ingresso uscita*	LEGGI A
• *Rettangolo: indica un'assegnazione cioè un'azione elementare, come assegnare un valore o il risultato di un calcolo ad una variabile*	A = 12
• *Rombo: con un ingresso e due uscite, indica una condizione in base al cui risultato, che può essere vero o falso, si decide il percorso da seguire*	Falso A > 10 Vero

La sintassi dei diagrammi di flusso prevede delle regole di base tra cui:

- Deve essere presente un solo blocco di inizio e almeno un blocco di fine
- Il blocco di inizio ha una sola freccia uscente
- Il blocco di fine ha una sola freccia entrante
- I blocchi di Input e Output e i blocchi di assegnazione hanno una sola freccia entrante e una sola freccia uscente
- Il blocco condizione ha una sola freccia entrante e due frecce uscenti contrassegnate dalle indicazioni vero e falso

Prima di vedere i diagrammi di flusso dobbiamo ancora definire alcuni concetti.

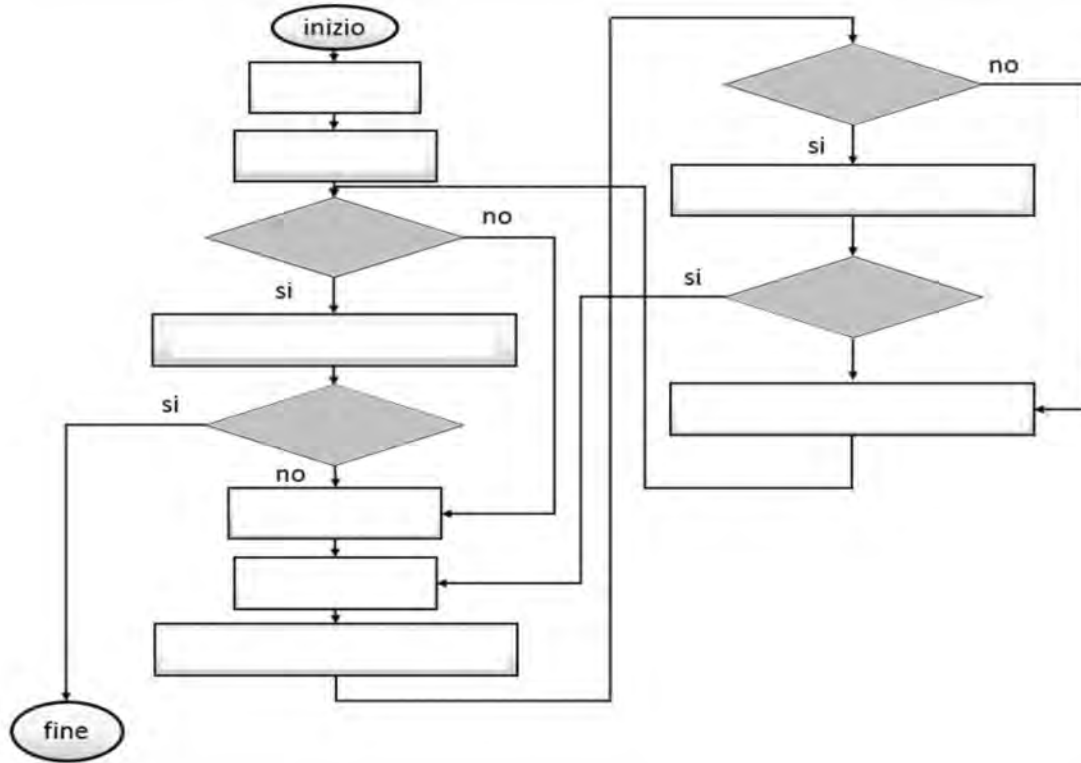
Variabili e istruzioni

Supponiamo che ci chiedano di fare una somma a mente, quando ci dicono il primo e poi il secondo numero li dobbiamo "ricordare", analogamente in un algoritmo bisogna ricordare i dati e lo si fa con l'uso delle variabili. Gli algoritmi operano essenzialmente su variabili che conterranno i dati sui quali si eseguiranno delle elaborazioni, quindi è necessario che i valori da elaborare vengano ***assegnati*** alle variabili prima di effettuare l'elaborazione. Per comprenderne meglio il significato possiamo pensare ad una variabile come ad una scatola identificata da un nome, se volessimo fare l'addizione di due numeri, potremmo chiedere di inserire il primo numero per esempio nella variabile A, poi il secondo numero nella variabile B, quindi possiamo fare l'addizione e inserirla nella variabile Somma, per capire quindi il significato di questo processo, rappresentiamo il tutto in un disegno:

I valori iniziali delle variabili possono derivare da :

- un'**istruzione di input**, per esempio se l'algoritmo deve calcolare l'area di un triangolo al suo interno saranno presenti due istruzioni di input che consentiranno all'utente di inserire i valori dall'esterno probabilmente tramite la tastiera, tali valori saranno conservati in due variabili per la successiva elaborazione.
- Un'**istruzione di assegnamento**, in cui è lo stesso algoritmo che assegna un determinato valore ad una variabile. Per esempio se volessimo calcolare la media di tre numeri chiesti all'utente potremmo definire la variabile media e assegnarle il valore iniziale zero, in questo modo si prepara la variabile per l'elaborazione o si conserva nella variabile un valore intermedio prodotto da una elaborazione precedente. Si può assegnare ad una variabile un valore costante come anche il valore risultante da una espressione aritmetica, come nell'esempio del disegno precedente dove a Somma viene assegnato come valore il risultato di A + B.

Le **istruzioni di output** si usano per comunicare all'esterno i valori delle variabili contenenti i risultati dell'elaborazione effettuata, anche qui si può far riferimento all'ultima parte del disegno precedente dove dalla scatola Somma eseguiamo un operazione di output mostrando il contenuto in quell'istante.

Strutture fondamentali e programmazione strutturata

Dopo aver descritto il lessico dei diagrammi di flusso cioè i simboli grafici con quali costruire i diagrammi, osserviamo la seguente figura:

Non abbiamo inserito volutamente comandi e istruzioni nei singoli blocchi perché il nostro scopo è evidenziare come, pur con le regole di sintassi finora descritte, possiamo concludere che la rappresentazione del diagramma a blocchi non è chiara, anzi sembra molto complessa, non si intuisce il flusso di esecuzione e, possiamo anticipare che una codifica dello stesso in un linguaggio di programmazione sarebbe complessa. Esiste una soluzione per ovviare a tale complessità e ai problemi ad essa connessi, è la **programmazione strutturata**. Disciplina nata alla fine degli anni 60, è una tecnica di progettazione di algoritmi che si basa sulle regole enunciate dal teorema di Böhm e Jacopini, che afferma che ogni *algoritmo è rappresentabile attraverso tre strutture fondamentali:* **sequenza**, **selezione** e **iterazione**. Si parla in questo caso di **programmazione strutturata**, questa tecnica consente di effettuare facilmente la codifica cioè la traduzione dal diagramma di flusso nel corrispondente programma scritto con un linguaggio di programmazione riducendo errori di progettazione e aumentando la "*leggibilità*" dell'algoritmo.

Sono disponibili dei software che consentono non solo la progettazione di algoritmi strutturati, ma anche la simulazione di esecuzione. Tra questi abbiamo Algobuild, e Flowgorithm, scansionando i seguenti codici QR, o utilizzando il relativo link, accederai ad un breve video che li descrive.

Algobuild
http://qrbridge.me/9n7b

Flowgorithm
http://qrbridge.me/9n7y

Nei prossimi paragrafi descriveremo le strutture fondamentali, progettando semplici algoritmi che li rappresentano, utilizzando come linguaggio di rappresentazione sia una Pseudo-linguaggio, quindi a caratteri, sia i diagrammi di flusso, quindi grafico.

La sequenza

La sequenza è la struttura più semplice, consiste in **un elenco ordinato di istruzioni**, ogni istruzione produce un risultato perché inserita in un contesto logico, che è quello determinato dalle istruzioni che la precedono.

Esempio: Algoritmo che calcola l'Area del triangolo, in questo algoritmo notiamo la successione di istruzioni, appunto la sequenza. (Ricevi intendiamo l'operazione di INPUT del dato dall'esterno e con Comunica l'operazione di Output del risultato verso l'esterno).

Rappresentazione con Pseudo-Linguaggio	Rappresentazione con Diagramma di Flusso generato col Software Gratuito Algobuild (www.algobuild.com)
1. INIZIO 2. Ricevi Base 3. Ricevi Altezza 4. Area ← (Base * Altezza) / 2 5. Comunica Area 6. FINE	START Triangolo IN Base IN Altezza Area = (Base * Altezza) / 2 OUT Area END Triangolo

La selezione

La selezione è una struttura che permette di **scegliere tra due alternative la sequenza di esecuzione.** Facciamo un esempio: vogliamo realizzare un algoritmo della divisione che richiede in ingresso il Dividendo e il Divisore, calcoli e poi visualizzi il Quoziente, sembrerebbe possibile utilizzare anche in questo caso la Sequenza:

1. INIZIO
2. Ricevi DIVIDENDO
3. Ricevi DIVISORE
4. QUOZIENTE= DIVIDENDO/DIVISORE
5. Comunica QUOZIENTE
6. FINE

L'algoritmo sembra funzionare correttamente ma se come Divisore venisse inserito il valore zero? La divisione sarebbe impossibile, quindi abbiamo due alternative, se il divisore è diverso da zero, eseguiamo il calcolo e mostriamo il risultato, altrimenti diamo il messaggio "Operazione impossibile". (si noti che nel linguaggio informatico diverso si scrive "!=" e si può leggere anche "non uguale").

Rappresentazione con Pseudo-Linguaggio	Rappresentazione con Diagramma di Flusso generato col Software Gratuito Algobuild (www.algobuild.com)
1. INIZIO 2. Ricevi DIVIDENDO 3. Ricevi DIVISORE 4. **SE** DIVISORE != 0 (diverso da 0) QUOZIENTE= DIVIDENDO/DIVISORE Comunica QUOZIENTE **ALTRIMENTI** Comunica "Operazione impossibile" **FINE SE** 5. FINE	START Divisione IN Dividendo IN Divisore If Divisore ! = 0 F T OUT "Operazione impossibile" Quoziente = Dividendo/Divisore OUT Quoziente END Divisione

La condizione espressa nella struttura "Se" permette di scegliere, in relazione al valore di verità o falsità, quale elaborazione svolgere. Questa struttura corrisponde nello pseudo-linguaggio ad un "SE condizione ALLORA … ALTRIMENTI … " ; non è necessario che nel ramo falso ci siano istruzioni da eseguire, in questo caso la struttura corrisponde ad un "SE condizione ALLORA…".

L'iterazione o ciclo

Spesso per risolvere dei problemi bisogna ripetere le stesse sequenze di azioni più volte, fino a quando non si verifica una condizione che fa terminare le operazioni. Supponiamo di voler dare le istruzioni per voler disegnare su un foglio a quadretti un quadrato di lato 5 quadretti, fissato il punto di partenza, supponiamo inoltre di avere a disposizione per muovere la penna i seguenti comandi: Avanti(n), dove n è il numero di quadretti, Destra(g) e Sinistra(g) dove g è la misura in gradi della direzione della penna, allora il nostro algoritmo dovrebbe essere costituito dalle seguenti istruzioni:

1. Avanti(5)
2. Destra(90)
3. Avanti(5)
4. Destra(90)
5. Avanti(5)
6. Destra(90)
7. Avanti(5)
8. Destra(90)

Abbiamo dovuto definire 8 istruzioni per risolvere il nostro l'algoritmo; se avessimo avuto a disposizione l'istruzione del tipo Ripeti(x) volte {le istruzioni del ciclo} il nostro algoritmo si sarebbe ridurre alla definizione di 3 istruzioni:

1. Ripeti (4)
2. {Avanti(5)
3. Destra(90)}

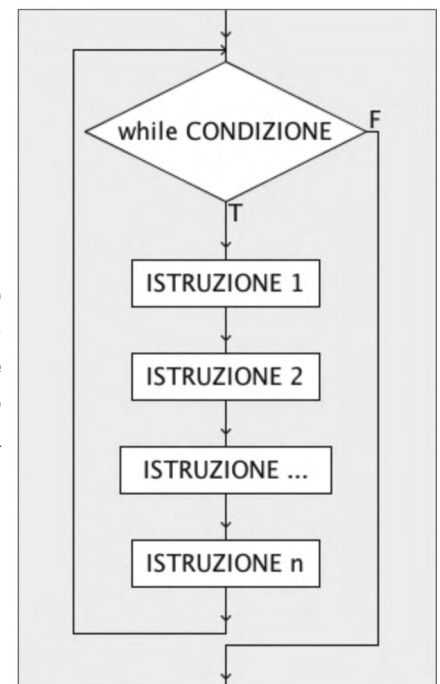
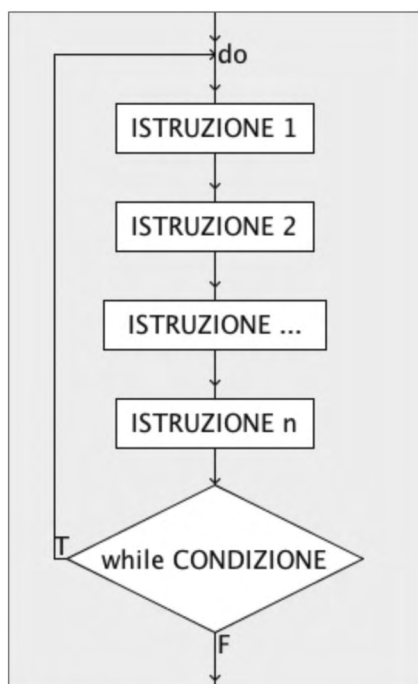
Questo è un esempio che mette in evidenza il vantaggio di poter definire dei comandi in strutture cicliche, anche se questo è solo un caso particolare, quello cioè in cui è noto a priori quante volte eseguire un ciclo, in questo caso quattro. Molto più spesso vi sono problemi in cui le istruzioni del ciclo si devono ripetere fino a quando non si verifica una particolare condizione, per esempio se volessimo tracciare una linea che va da un bordo all'altro della pagina senza sapere da quanti quadretti è composta la larghezza della pagina nella progettazione dell' algoritmo abbiamo altre possibili soluzioni come la seguente

Finchè non raggiungi il Bordo
{Vai Avanti(1)}

in questo caso la condizione di uscita dal ciclo è aver raggiunto il bordo.
A questo punto possiamo definire formalmente quali sono le strutture iterative previste nella programmazione strutturata:

1. **WHILE** (Condizione)
{ ISTRUZIONE 1
ISTRUZIONE 2
ISTRUZIONE ...
ISTRUZIONE n}

While ... Do

Chiamato anche While Do, in italiano tradotto in Mentre (condizione) {istruzioni…} Fine_Mentre è caratterizzato dal fatto che il punto di ingresso è proprio il blocco di controllo della condizione, cioè il controllo avviene **in testa al ciclo e itera (cicla) per vero**, (T in figura significa True) cioè le istruzioni del ciclo vengono eseguite se la condizione è vera, si esce dal ciclo quando il controllo della condizione da esito falso; ciò comporta il caso limite che se al primo test della condizione questa risulta subito falsa le istruzioni del ciclo non vengono eseguite neanche una volta.

Do .... While

2. **DO**
{ ISTRUZIONE 1
ISTRUZIONE 2
ISTRUZIONE ...
ISTRUZIONE n}

WHILE (Condizione)

Chiamato anche Repeat…Until in alcuni linguaggi di programmazione, in italiano è stato tradotto in Ripeti {istruzioni…} Finchè (condizione), è caratterizzato dal fatto che le istruzioni del ciclo si trovano posizionate prima del blocco di controllo della condizione, anche questa struttura **itera (cicla) per vero** come la precedente ed il controllo avviene **in coda al ciclo**, quindi prima del controllo della condizione le istruzioni del ciclo vengono eseguite una prima volta, numero minimo di esecuzioni delle istruzioni del ciclo qualora la condizione risultasse subito falsa.

Esempio: progettare l' algoritmo *che ripete ciclicamente la richiesta di un numero in ingresso, calcola e visualizza il suo quadrato, fino a quando il numero inserito è maggiore di zero, se viene inserito 0 l'algoritmo termina.* Si tratta in altri termini di effettuare la stessa elaborazione, cioè calcolo e visualizzazione del quadrato di un numero, effettuata su numeri diversi, quelli che arriveranno dall'input:

1. Inizio
2. Ricevi NUMERO
3. Mentre NUMERO > 0
4. Assegna a QUADRATO valore NUMERO*NUMERO
5. Comunica QUADRATO
6. Ricevi NUMERO
7. Fine-mentre
8. Fine

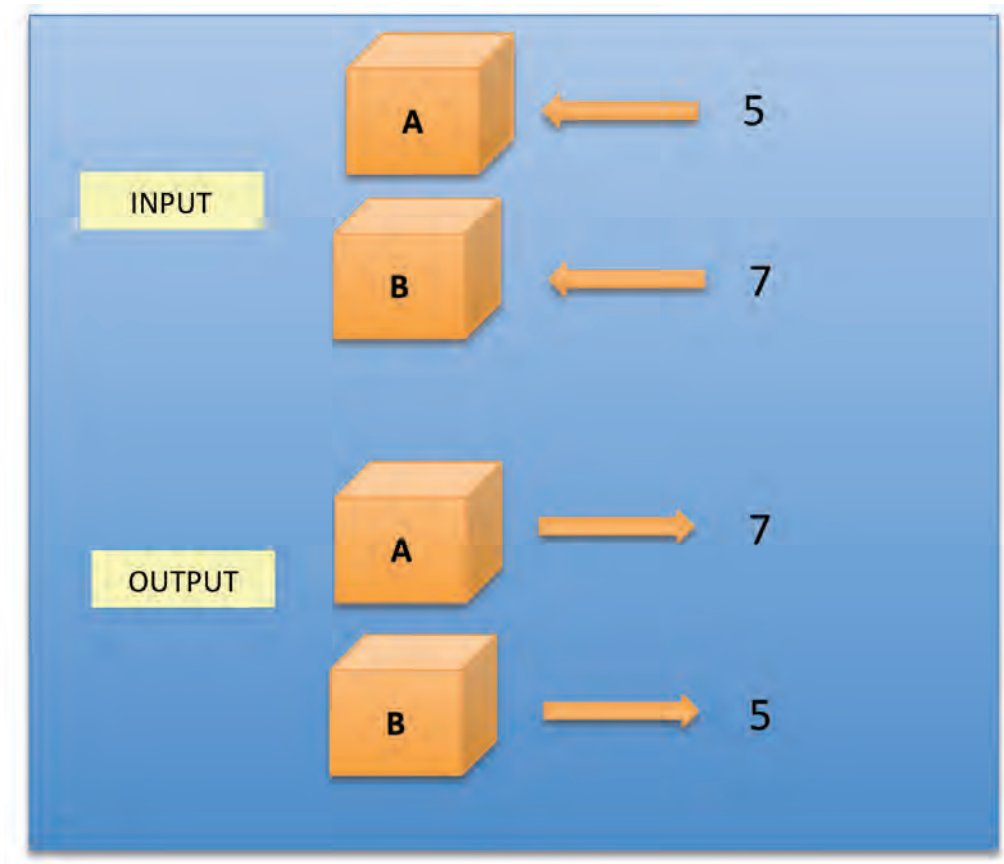
Dentro la struttura iterativa, fra le parole Mentre e Fine-mentre sono definite le istruzioni per il calcolo del quadrato del numero: il ciclo permette di ripetere tale calcolo per ogni numero che verrà dato in input nell'istruzione numero 5. La condizione NUMERO>0 viene chiamata **condizione di controllo del ciclo**, se il valore introdotto in input è non positivo l'esito del controllo sarà Falso e quindi si uscirà dal ciclo. È necessario inserire un primo input fuori ciclo per permettere superare il primo controllo della condizione del ciclo stesso.

Rappresentazione con Pseudo-Linguaggio	Rappresentazione con Diagramma di Flusso generato col Software Gratuito Algobuild (www.algobuild.com)
1. INIZIO 2. Ricevi NUMERO 3. **MENTRE** NUMERO > 0 QUADRATO = NUMERO* NUMERO Comunica QUADRATO Ricevi NUMERO **FINE MENTRE** 4. FINE	START Ciclo IN Numero while Numero > 0 F T Quadrato = Numero * Numero OUT Quadrato IN Numero END Ciclo

Attenzione: Sebbene quasi tutta la bibliografia definisce le strutture iterative nello stesso modo fin qui descritto è possibile trovare fonti anche autorevoli in cui le stesse strutture utilizzano valori di verità per ripetere o uscire dal ciclo inversi, ciò porta alla conclusione che le strutture iterative possono essere considerate quattro, cioè le due illustrate che poi si possono distinguere ulteriormente in due versioni, in base a come si dispongono i valori di verità sui rami d'uscita del blocco di controllo.

Le tre strutture fondamentali descritte, possono essere ricombinate più volte, ad esempio all'interno di una sequenza possiamo inserire iterazioni e selezioni che, a loro volta, possono contenere sequenze, selezioni e iterazioni. Il rispetto rigido delle strutture consente una traduzione nel linguaggio di programmazione abbastanza semplice e di mantenere "leggibilità" dell'algoritmo notevole. Le strutture fondamentali possono essere paragonate ai famosi mattoncini giocattolo le cui combinazioni permettono la costruzione di oggetti di varia complessità.

Esempi fondamentali: Lo scambio e la variabile temporanea

Un esempio classico nella progettazione di algoritmi è quello in cui dati in input due valori a due variabili si richiede che in output le variabili contengano l'una il valore dell'altra e viceversa, graficamente possiamo rappresentare la situazione come segue:

Facciamo una precisazione, secondo il significato delle istruzioni nel linguaggio informatico, ciò che si trova a destra dell'operatore di assegnazione che è definito sorgente viene assegnato a ciò che si trova a sinistra dell'operatore che è definito destinazione. L'operatore di assegnazione è di solito rappresentato con il simbolo "=" o con una freccia orientata verso sinistra "←", la destinazione deve essere per forza una variabile, la sorgente può essere un valore, un'altra variabile o un espressione, Quindi il significato delle istruzioni di assegnazione è del tipo:

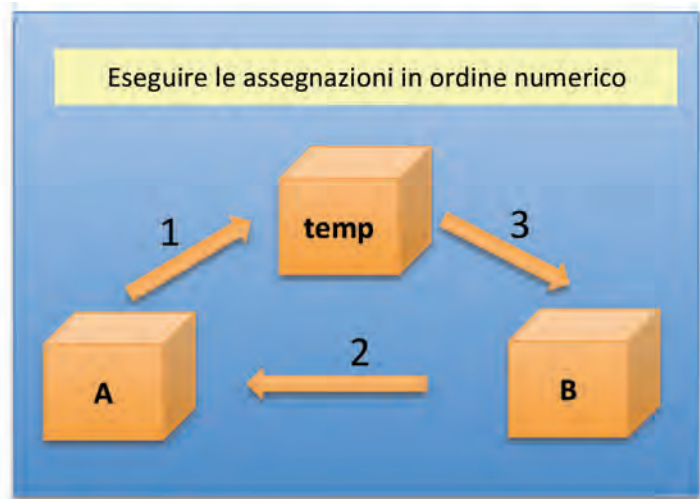
Destinazione ← Sorgente

in pratica se scrivo

Area = b*h/2

dove b ed h contengono dei valori precedentemente inseriti, alla variabile Area verrà assegnato il risultato dell'espressione aritmetica. Tornando al nostro problema se procediamo nel seguente modo:

	prima	*dopo*
A ← B	*A=5; B=7;*	*A=7;B=7;*
B ← A	*A=7;B=7;*	*A=7;B=7;*

Questo perché nella prima assegnazione il valore della destinazione viene sovrascritto da quello della sorgente quindi viene distrutto. Il problema si risolve con l'uso di una nuova variabile detta d'appoggio o temporanea, che ha lo scopo di conservare temporaneamente il valore in questo caso di A, prima che venga distrutto, vediamo graficamente come si procede chiamando la nostra variabile temp:

Quindi si procede così:

1	*temp ← A*	*A=5; B=7; temp =5;*
2	*B ← A*	*A=7; B=7; temp =5;*
3	*B ← A*	*A=7; B=5; temp =5;*

Rappresentazione con Pseudo-Linguaggio	Rappresentazione con Diagramma di Flusso generato col Software Gratuito Algobuild (www.algobuild.com)
1. INIZIO 2. Ricevi A 3. Ricevi B 4. temp = A 5. A = B 6. B = temp 7. comunica A 8. comunica B	START Scambio ↓ IN A ↓ IN B ↓ temp = A ↓ A = B ↓ B = temp ↓ OUT A ↓ OUT B ↓ END Scambio

Esempi fondamentali: La variabile accumulatore

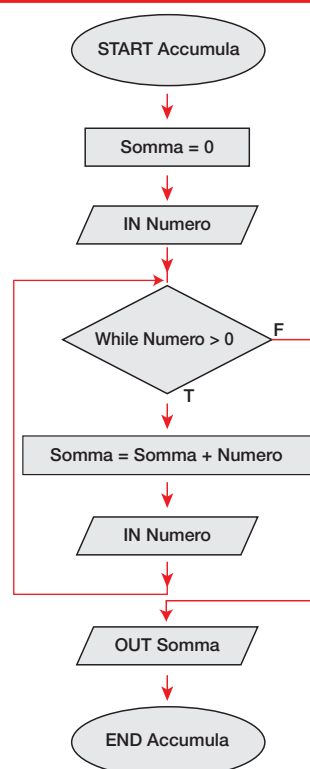
Molti algoritmi richiedono l'uso di variabili chiamate spesso accumulatori o totalizzatori per la loro funzione. Un esempio può essere la cassa di un supermercato, il cliente presenta i prodotti che vuole acquistare, il passaggio del codice a barre dal lettore ottico corrisponde all'inserimento del costo del prodotto che viene aggiunto all'importo totalizzato fino a quel momento, si tratta cioè di un ciclo in cui per ogni oggetto viene acquisito il prezzo e accumulato in modo da stabilire l'importo totale della spesa. Se volessimo progettare l'algoritmo di questa funzione utilizzeremmo una variabile che rappresenta il totalizzatore di cassa, che potremmo chiamare Importo, la quale, partendo dal valore iniziale zero viene aggiornata aggiungendo al suo valore il prezzo di ogni nuovo prodotto inserito. Al termine dell'algoritmo Importo conterrà il valore totale da corrispondere.

La variabile Importo nell'esempio è quella che, nel linguaggio della programmazione, viene definita **totalizzatore** o **accumulatore**: cioè una variabile nella quale ogni nuovo valore si somma a quelli già presenti in precedenza. Vediamo un altro esempio di uso della variabile accumulatore:

Esempio: Esegui la somma di una serie di numeri, la serie termina inserendo il numero zero e mostra il risultato

1. INIZIO
2. **INIZIALIZZA** SOMMA con valore 0
3. Ricevi NUMERO
4. **MENTRE** NUMERO >0
   SOMMA =SOMMA + NUMERO
   Ricevi NUMERO

FINE MENTRE

5. **COMUNICA** SOMMA
6. FINE

In questo esempio è necessario effettuare l'operazione di **inizializzazione della variabile**, La variabile accumulatore SOMMA è inizializzata, prima del ciclo, al valore zero, perché non essendo ancora stati inseriti dei numeri non può avere valore diverso e ancor di più perché non sarebbe possibile infatti eseguire correttamente l'azione di aggiungere un numero al valore di SOMMA, se tale valore non esiste, per un esecutore come il computer questa situazione causerebbe un errore. Al termine dell'algoritmo abbiamo l'operazione di output di SOMMA, per mostrare il risultato ottenuto.

Esempi fondamentali: La variabile contatore

Il **contatore** è una variabile simile alla precedente, ma con la caratteristica che si aggiorna sempre di una quantità costante. Pensiamo per esempio all'esercizio fisico delle flessioni, quando l'esecuzione consiste nel ripetere gli stessi movimenti coordinati incrementando ogni volta di uno il nostro conteggio, il valore uno è detto **passo d'incremento** o **step**, potremmo per esempio decidere di contare partendo dal valore dieci fino ad arrivare a zero, in questo caso lo step è -1; potremmo contare il nostro esercizio non ad ogni flessione ma al raggiungimento di ogni decina, cioè 10, 20… in questo caso lo step è 10. Vediamo un esempio di uso della variabile contatore:

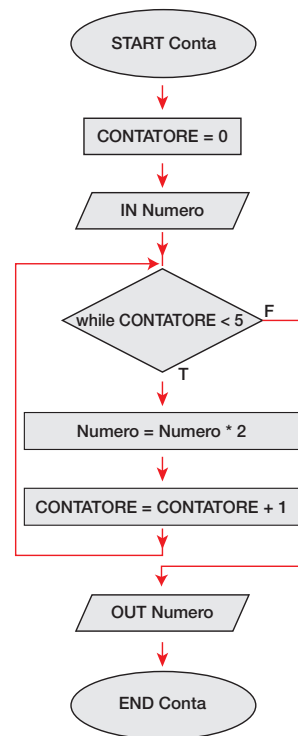
Esempio uso del contatore: Esegui cinque volte il doppio di un numero inserito ed infine mostra il risultato

1. INIZIO
2. INIZIALIZZA CONTATORE con valore 0
3. Ricevi NUMERO
4. **MENTRE** CONTATORE < 5

   NUMERO = NUMERO*2

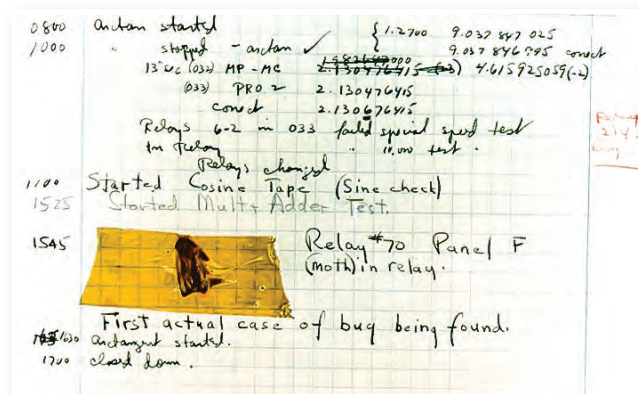
   CONTATORE = CONTATORE + 1

FINE MENTRE

5. COMUNICA NUMERO
6. FINE

In questo esempio è necessario effettuare l'operazione di inizializzazione della variabile CONTATORE in modo che alla prima esecuzione dell'istruzione di aggiornamento CONTATORE=CONTATORE+1 sia noto a quale valore iniziale eseguire l'incremento.

Debug e correzione di errori: la tecnica della trace table

Nel progettare algoritmi è molto probabile che si commettano degli errori, spesso non immediatamente visibili, ma rilevabili solo dopo diverse simulazioni di esecuzione, in particolare introducendo come valori d'ingresso i cosiddetti "casi limite"; abbiamo visto l'esempio della divisione, il caso limite era l'introduzione del valore zero come Divisore, risolto appunto con l'introduzione della struttura di selezione.

L'attività di ricerca di errori negli algoritmi ed in generale nei programmi ha un nome ormai quasi di uso comune nelle nuove generazioni: Debugging; Non è raro sentire l'affermazione "è buggato" riferito a qualcosa che non funziona, il termine nasce dalla parola Bug, ed riferito all'insetto, precisamente una falena, che nel 1947 causò un malfunzionamento del computer MARK II, di cui abbiamo parlato nel primo capitolo. Nel registro cartaceo, il tenente Grace Hopper capo del gruppo responsabile del calcolatore, annotò "…First actual case of bug being found".

Registro eventi Mark II conservato al Smithsonian National Museum of American History (Washington)

Una tecnica semplice ed efficace di simulazione di funzionamento dell'algoritmo è la Tabella di traccia" (o trace table), che viene costruita riportando su di essa ad ogni esecuzione di un istruzione una nuova riga che contiene, i cambiamenti dei valori delle variabili o gli esiti dei controlli di condizione, nella stessa tabella vengono anche rappresentati i valori emessi in uscita dall'algoritmo. Costruiamo a titolo di esempio la trace table per l'ultimo algoritmo mostrato, cioè: "Esegui la somma di una serie di numeri, la serie termina inserendo il numero zero e mostra il risultato", il cui algoritmo in Pseudo-Linguaggio è il seguente:

```
1.  INIZIO
2.  INIZIALIZZA SOMMA con valore 0
3.  Ricevi  NUMERO
4.  MENTRE   NUMERO >0
5.    SOMMA =SOMMA + NUMERO
6.    Ricevi  NUMERO
7.  FINE MENTRE
8.  COMUNICA SOMMA
9.  FINE
```

Nella simulazione ipotizziamo di Inserire ad esempio la sequenza di numeri (5,7,3,0) e per semplicità omettiamo dalla tabella le istruzioni di inizio, Fine Mentre e Fine, che non producono variazioni. La tabella sarà la seguente:

Numero	*Istruzione*	*Somma*	*Numero*	*(Numero > 0)*	*OUTPUT*
1	*2*	*0;*			
2	*3*		*5*		
3	*4*			*VERO*	
4	*5*	*5*			
5	*6*		*7*		
6	*4*			*VERO*	
7	*5*	*12*			
8	*6*		*3*		
9	*4*			*VERO*	
10	*5*	*15*			
11	*6*		*0*		
12	*8*				*15*
					Fine elaborazione

Se per esempio avessimo sbagliato la condizione scrivendo “Numero < 0”, ci saremmo subito accorti dell’errore non entrando mai nel ciclo, oppure se non avessimo messo l’istruzione 3 “Ricevi Numero”, fuori dal ciclo, avremmo notato subito l’errore al primo controllo della condizione.

Conclusioni

Progettare algoritmi è un’abilità che si acquisisce con l’esperienza, come nello sport o nella musica, si può partire da zero, iniziare con semplici esercizi, allenarsi e nel tempo si possono incrementare i livelli di difficoltà; dagli algoritmi alla programmazione il passo è breve, si può iniziare subito con linguaggi visuali di programmazione come Scratch, che viene illustrato nel capitolo 10 di questo libro. Da qui si potrebbero aprire scenari inimmaginabili oggi, la robotica, l’industria 4.0, la stampa 3D, la realtà aumentata e il Web in tutte le sue forme più innovative, le App Mobile, sono i settori in maggior fermento, sempre regolati da algoritmi e offrono enormi possibilità a tutti di entrare da protagonisti in un futuro che è già presente.

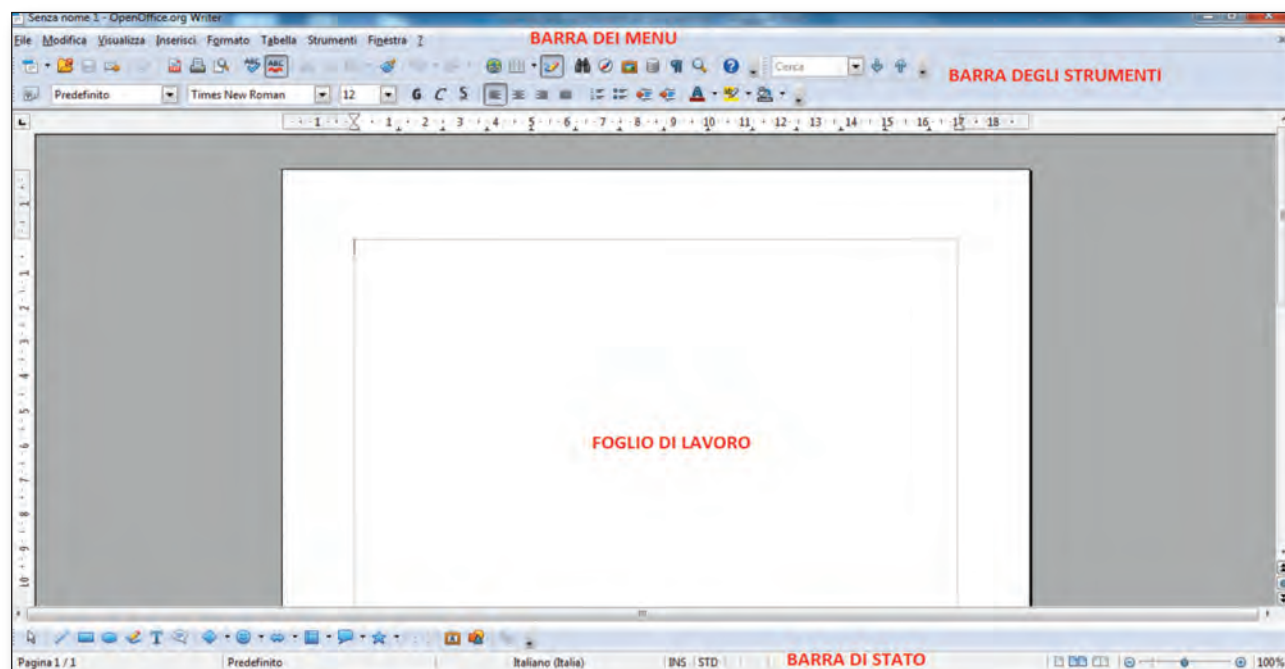
7. Gli elaboratori di testi (OpenOffice Writer)

Autori: **Giorgia Martina** e **Salvatore Madaro**

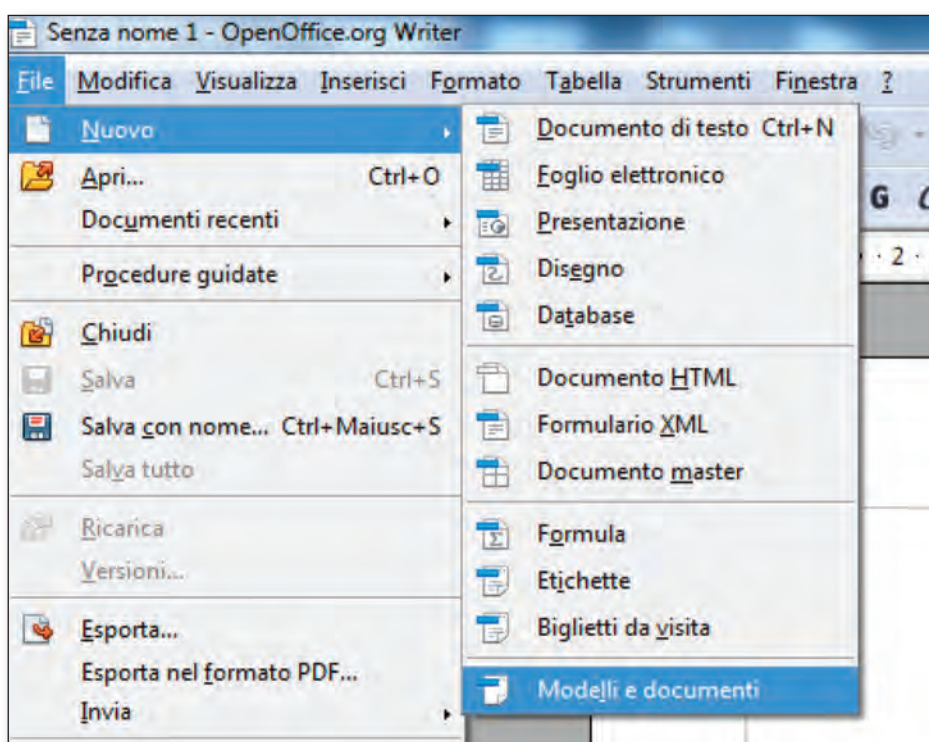
Competenze	Abilità	Conoscenze
Utilizzare, con autonomia operativa ed organizzativa, strumenti di comunicazione visiva e multimediale, anche con riferimento alle strategie espressive e agli strumenti tecnici della comunicazione in rete.	*Utilizzare i principali software per la produttività individuale.* *Raccogliere, organizzare e rappresentare informazioni.*	*Software di utilità e software applicativi.*

Tutte le attività che fanno riferimento alla scrittura, modifica, elaborazione, memorizzazione e stampa di documenti sono riconducibili ai cosiddetti software di videoscrittura.
In commercio, ma anche come opensource scaricabile da internet, esistono vari software che realizzano tutte queste attività. Anche se in parte esistono alcune differenze sia per quanto riguarda l'interfaccia grafica e sia per quanto riguarda i vari comandi, si possono comunque definire delle regole e dei concetti di base che sono comuni a tutti i software della stessa categoria e che devono essere noti prima dell'utilizzo; in questi appunti ci riferiamo al programma Writer di OpenOffice.

Prima di iniziare...

L'interfaccia grafica del programma

In generale, all'apertura del programma viene presentata una videata caratterizzata:

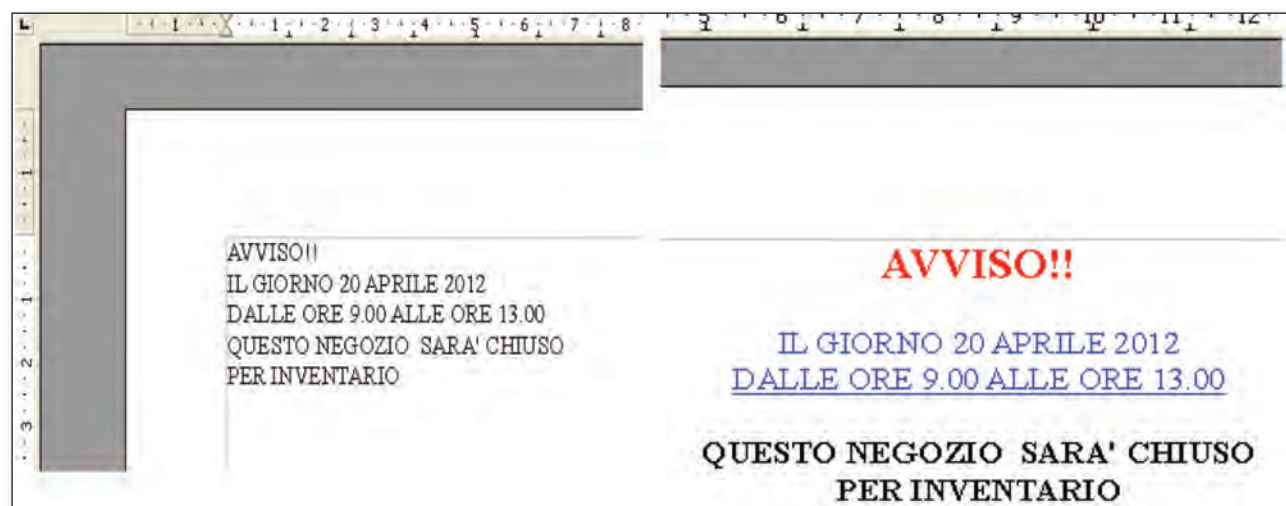
- da una parte alta nel quale si trovano sempre la "Barra dei Menu" e la "Barra degli strumenti"
- da un'area centrale completamente bianca che rappresenta un foglio sul quale l'utente dovrà scrivere il testo
- una parte bassa con altre "Barre degli strumenti" e una "Barra di stato".

Barra dei menu: è una riga che inizia sempre con “FILE” e finisce con “?” e in ogni voce vengono racchiusi tutti i possibili comandi che possono essere utilizzati all'interno del programma stesso.
Barra degli strumenti: sono un'insieme di icone ognuna delle quali rappresenta un comando specifico.
Normalmente non vengono visualizzati tutti i comandi ma l'utente può in qualsiasi momento decidere di attivare o disattivare specifici gruppi di comandi.
Barra di stato: indica in qualsiasi momento il numero di pagine, il tipo di foglio, la lingua utilizzata ed eventuali comandi attivati.
Sul lato sinistro e alto del foglio bianco viene indicato un righello che permette all'utente si sapere in quale parte del foglio sta scrivendo (margini del foglio).
Normalmente questo tipo di programma viene utilizzato per creare un documento partendo da un foglio nuovo, ma bisogna sapere che tutti i programmi contengono una serie di documenti-tipo (cosiddetti modelli) che, avendo già un contenuto di base specifico, danno la possibilità all'utente di modificare solo alcune parti e quindi di accelerare la produzione del documento stesso (ad esempio se si vuole creare un documento che dovrà essere inviato via fax, esistono dei modelli dove l'utente dovrà solo inserire il mittente, destinatario e contenuto ed il documento è fatto!).

Ogni documento prodotto deve essere salvato associando ad esso un nome. L'estensione (costituita di solito da 3 lettere) viene assegnata automaticamente dal programma. Spesso un documento creato con un programma dovrà essere aperto con un altro programma dello stesso tipo.

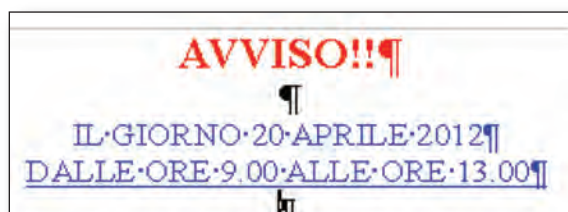
Ad esempio se si crea un documento con OpenOffice, questo si può aprire apre successivamente anche con Microsoft Office. In questi casi bisogna ricordarsi sempre di salvare il documento facendo riferimento al tipo di estensione del programma che dovrà aprirlo. Si ricorda che tutti i documenti creati con OpenOffice hanno estensione “.odt”, tutti i documenti creati con MicrosoftOffice hanno estensione “.doc” oppure “.docx”.

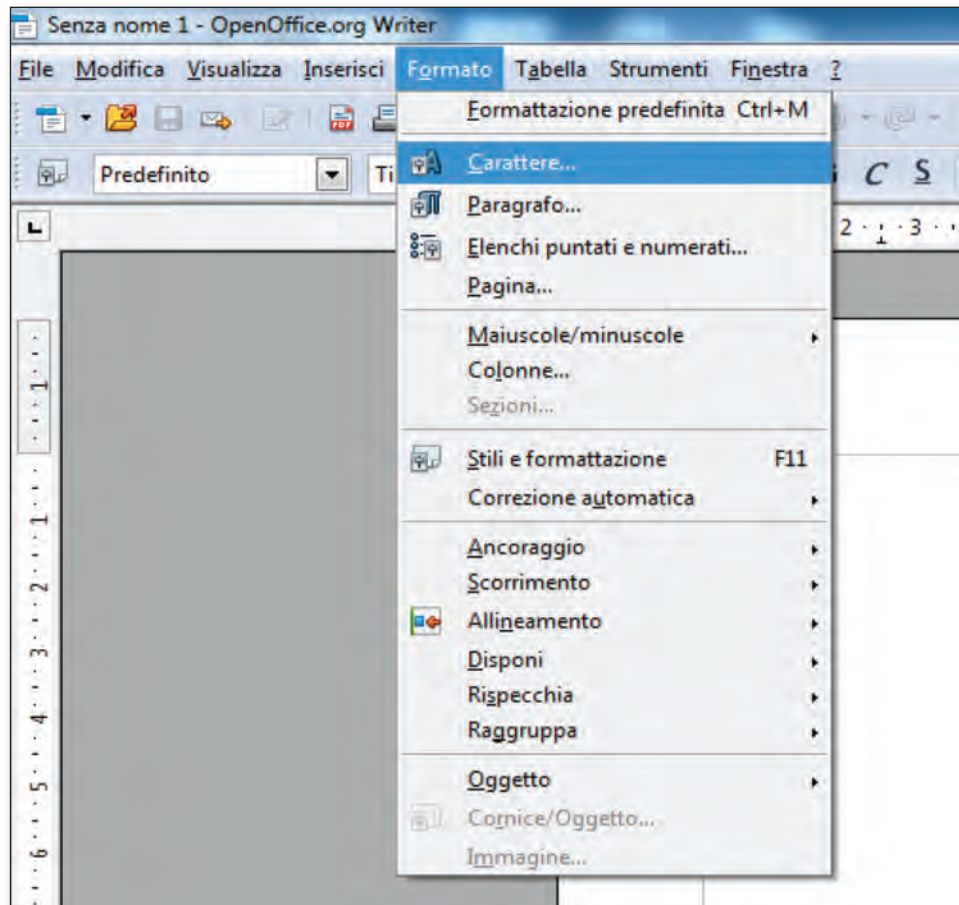
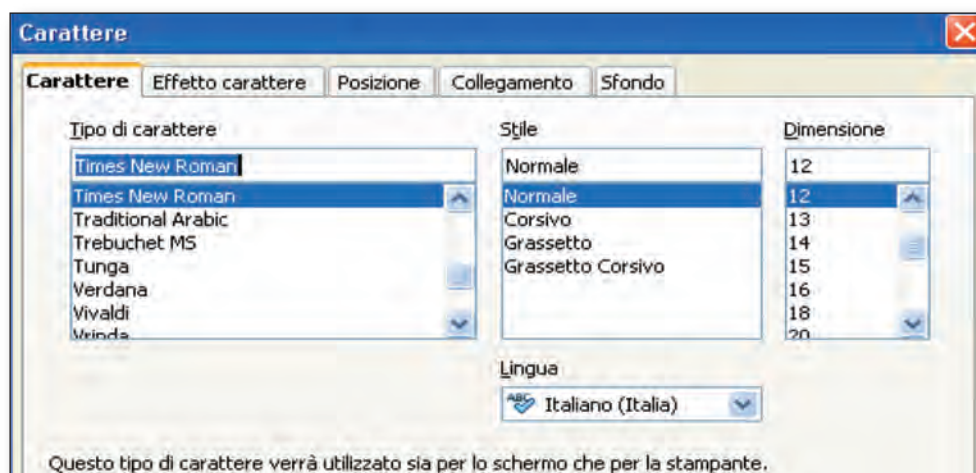
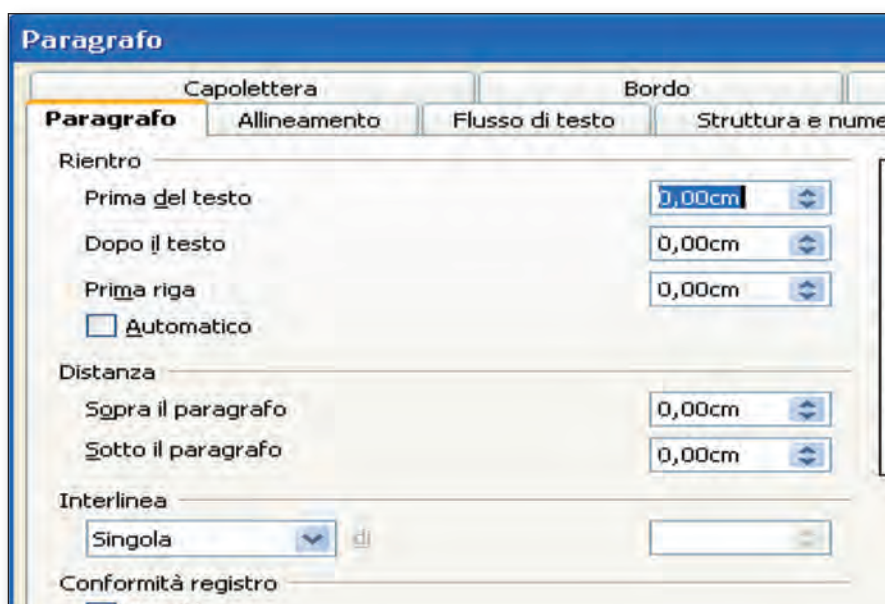
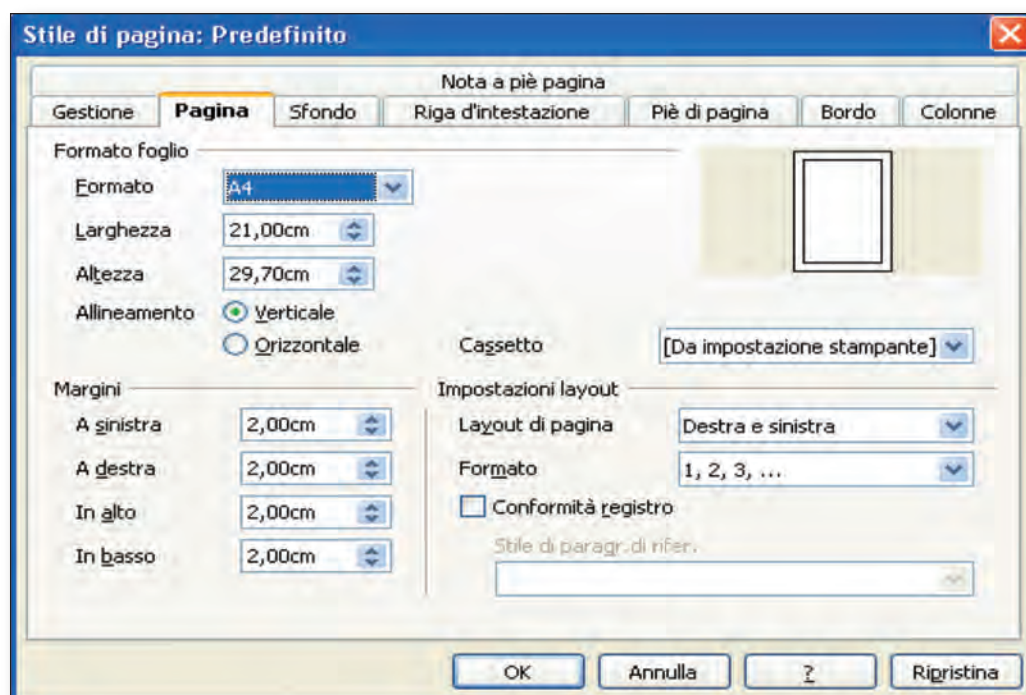
Creare un documento nuovo

Quando si usano questi programma è buona norma scrivere tutto il testo senza attivare alcun comando di formattazione, tenendo presente solo l'eventuale testo a capo (tasto Invio). Una volta che il testo è stato completamente scritto, si potranno utilizzare tutti i comandi di formattazione che permetteranno all'utente di realizzare un documento finale che abbia un layout, cioè un aspetto grafico, che renda la comunicazione del messaggio chiara ed efficace.

Esempio: **Testo non formattato** **Testo formattato**

Poco chiaro **Molto più chiaro**

Precisiamo che, all'interno di un documento scritto in un elaboratore testi
la parola è una sequenza di caratteri adiacenti
Il rigo è l'insieme di tutte le parole che si trovano su una linea.
Il paragrafo è l'insieme di tutte le parole che inserisco fino a quando si preme il tasto INVIO; Il fine paragrafo viene indicato con il simbolo con "¶" che può essere visualizzato assieme ad altri caratteri nascosti premendo il pulsante: ¶ .

Ad esempio:

Formattare il testo

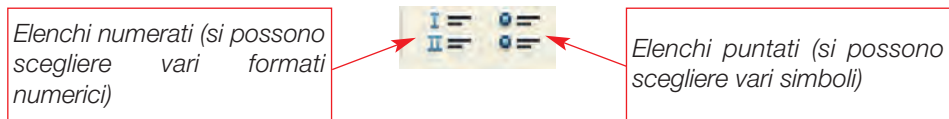
Quando si parla di formattazione del testo, ci si riferisce a tutti i comandi possibili che permettono si modificare le caratteristiche sia del carattere che del paragrafo; tali funzioni sono accessibili dalla barra degli strumenti oppure dalla barra dei menu alla voce “formato”

Dalla voce **carattere** si possono cambiare: tipo di scrittura, stili, dimensione e posizione

Dalla voce **paragrafo** si può cambiare la posizione del testo all'interno del foglio (ad esempio allineamento centrato, a destra, a sinistra oppure giustificato. Tra i comandi legati al paragrafo troviamo anche: interlinea (spazio tra un rigo ed un altro), rientro (spostamento del testo a destra o a sinistra rispetto alla marginatura del foglio) e lo spazio tra i paragrafi.

Dalla voce **Pagina** si possono cambiare: orientamento del foglio (orizzontale o verticale), dimensione della carta (es. A4), marginatura del foglio (sinistra, destra, alto, basso)

Una funzionalità del programma molto utile ed interessante è la possibilità, in qualsiasi momento, di spostare o copiare parte del testo da un punto ad un altro del documento creato oppure tra documenti dello stesso tipo oppure tra documenti di tipo diverso. Queste operazioni possono es sere eseguite grazie ai comandi **TAGLIA – COPIA – INCOLLA** che possiamo

trovare sulla barra degli strumenti con le seguenti icone:

oppure alla voce di **menu → MODIFICA**.

Per realizzare un documento ancora più completo è possibile aggiungere al testo degli oggetti aggiuntivi che possono essere: tabelle, grafici, immagini, disegni , espressioni matematiche particolari, foto e altro ancora. Tutti questi oggetti si trovano alla voce di **menu → INSERISCI**

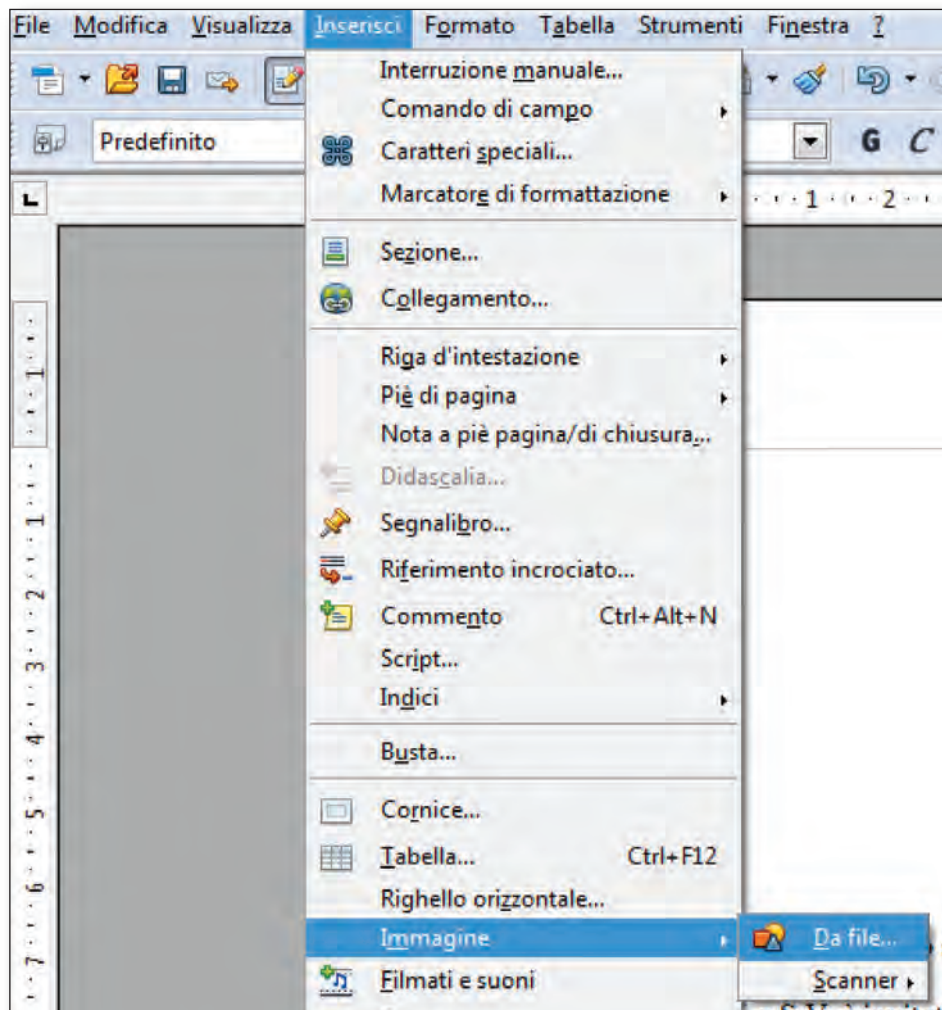
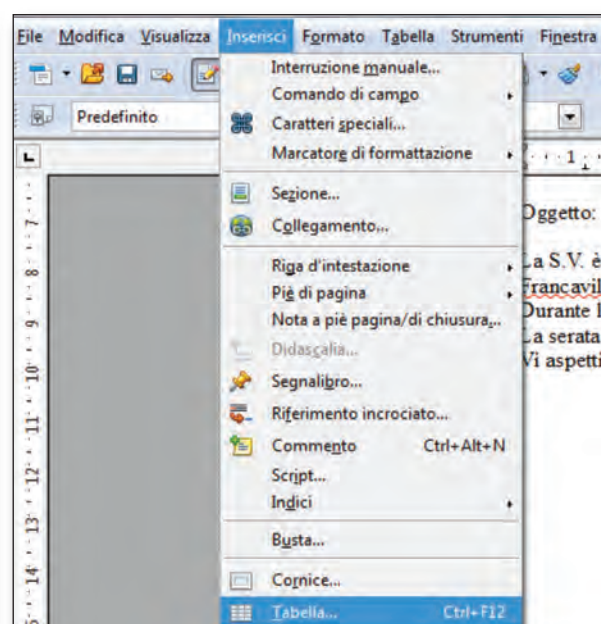
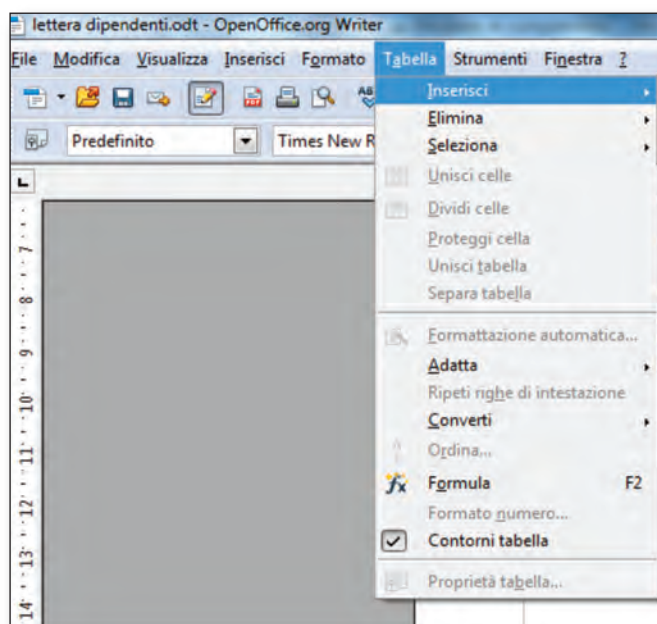
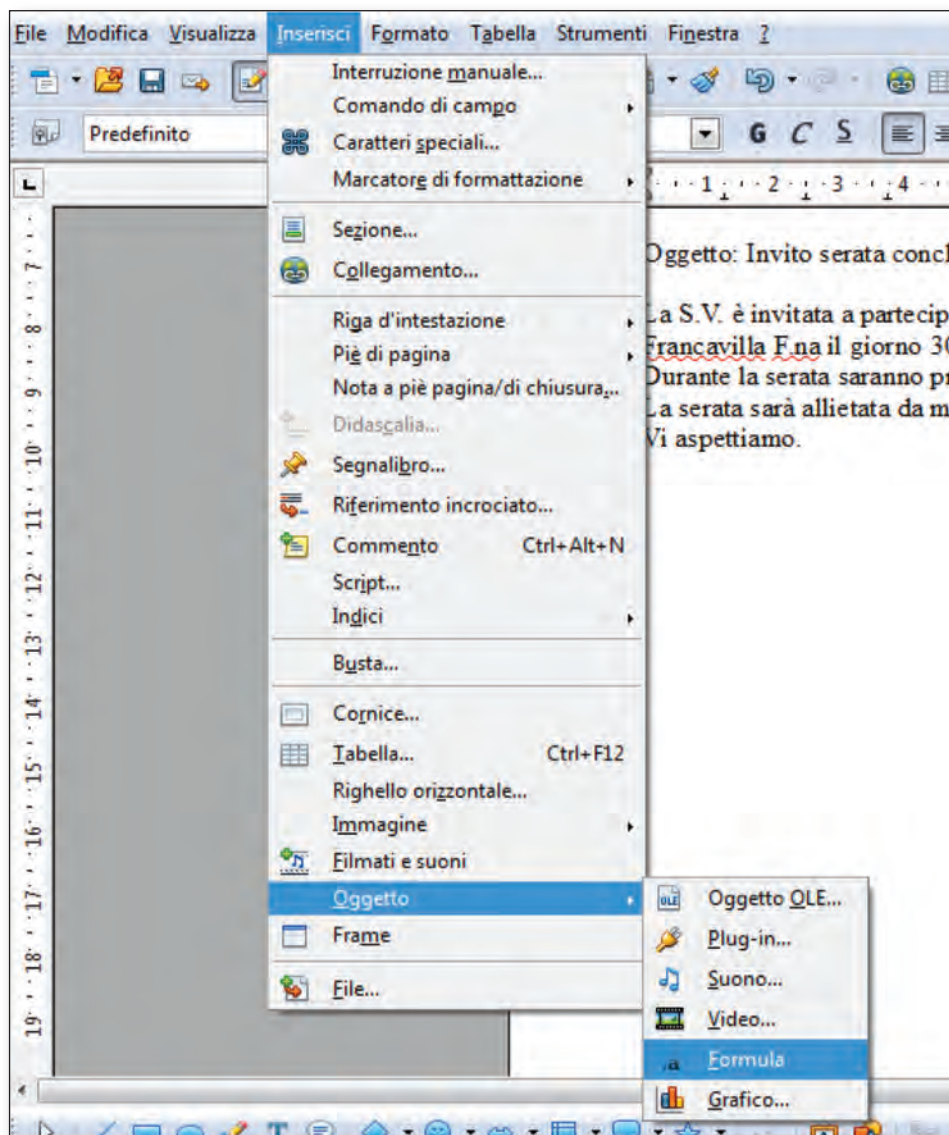
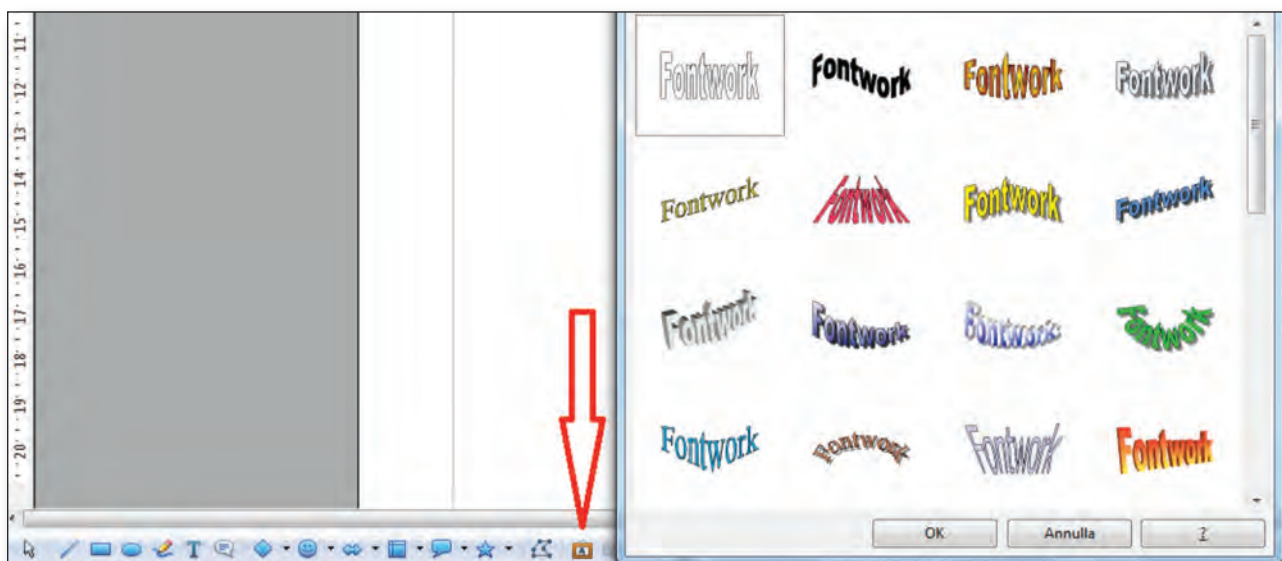
Gli elenchi puntati e numerati

È possibile creare elenchi pontati e numerati sia dalla barra degli strumenti

che dalla Barra di Menu:
selezionare la voce **"FORMATO → Elenchi puntati e numerati".**
Spesso succede di dover elencare, all'interno di un testo, alcune informazioni.

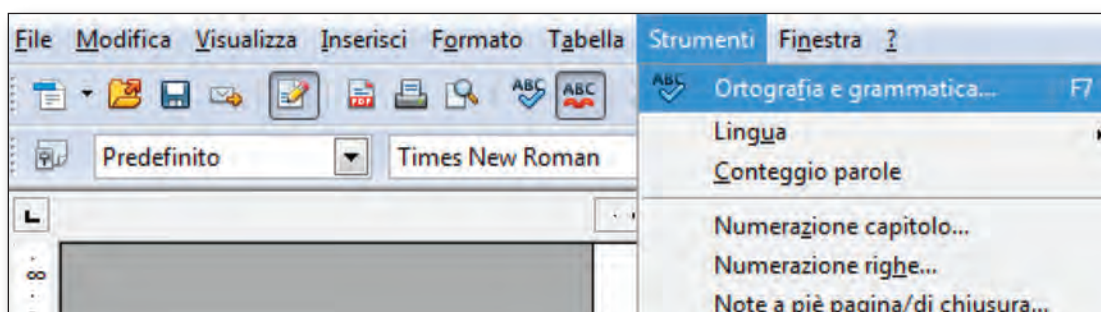
Esempio:

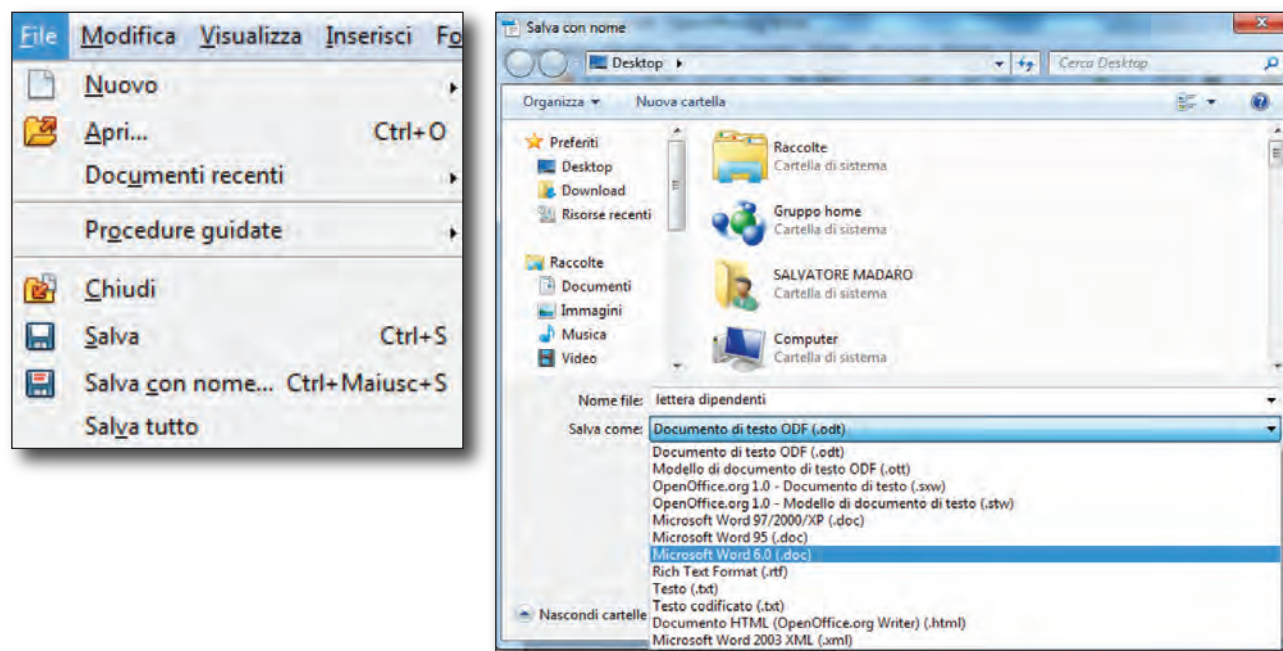
TESTO NON FORMATTATO	TESTO FORMATTATO
strutturazione dell'esame ECDL in moduli	**Strutturazione dell'esame ECDL in moduli**
Concetti di base dell'ICT	1. Concetti di base dell'ICT
Uso del computer	2. Uso del computer
Elaborazione testi	3. Elaborazione testi
Foglio elettronico	4. Foglio elettronico
Uno delle basi di dati	5. Uno delle basi di dati
Strumenti di presentazione	6. Strumenti di presentazione
Navigazione Web e Comunicazione	7. Navigazione Web e Comunicazione

Inserimento di oggetti

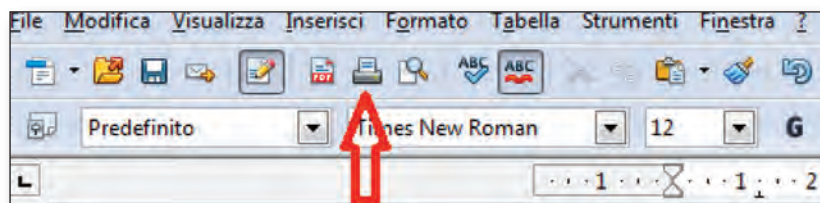
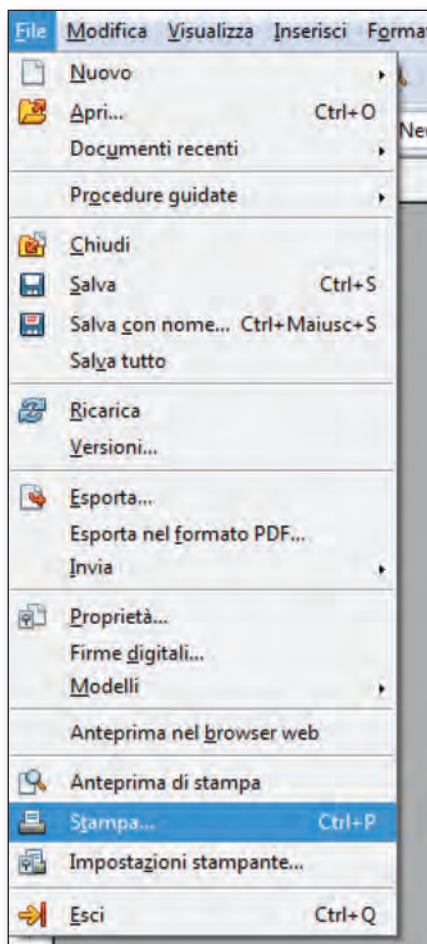
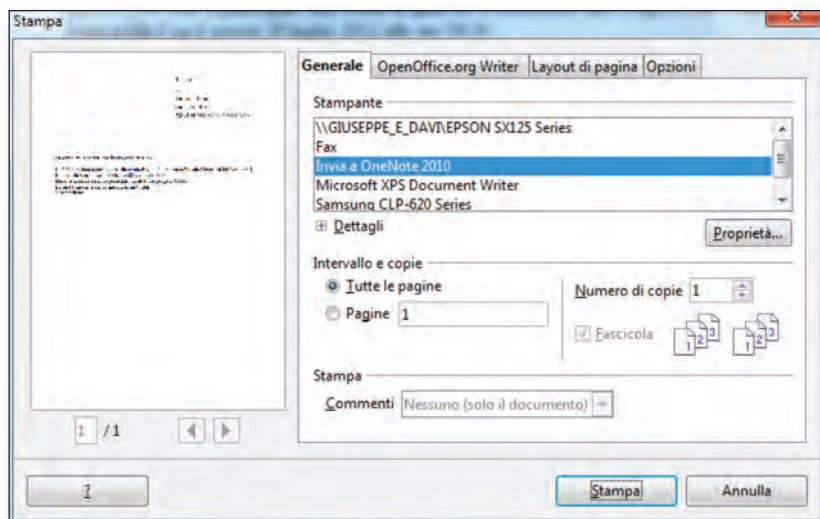
Per rendere più accattivante un documento è possibile inserirvi delle immagini:

delle tabelle (che rendono più ordinato un gruppo di dati)

delle tabelle (che rendono più ordinato un gruppo di dati)

del testo artistico, dei simboli, delle forme predefinite o dei semplici disegni a mano libera:

Questi ultimi elementi si inseriscono agendo sulla barra degli strumenti “Disegno”.

Ortografia

Prima di stampare il documento o di inviarlo a qualche altra persona è opportuno controllare l'ortografia per correggere eventuali errori:

Salvataggio dei dati

Il documento va salvato di tanto in tanto per evitare che per qualche motivo si perda quanto è stato scritto. Il programma permette di salvare il documento in diversi formati per consentire lo scambio di dati con altri programmi simili:

Il documento può essere salvato anche in formato PDF con il pulsante:

La stampa del documento

Dal menu dei comandi alla voce **"File"** oppure direttamente dalla barra degli strumenti è possibile eseguire la stampa del documento.

La differenza tra i due metodi di stampa è che mentre con la barra degli strumenti si ottiene una stampa immediata sulla stampante predefinita, attraverso il menu dei comandi è possibile scegliere varie opzioni, tra le quali la stampante di output, il numero di copie del documento, le pagine da stampare, la fascicolazione, il numero di pagine per foglio ecc..

Le intestazioni e i pie di pagina

Spesso, all'interno di un documento scritto, è necessario duplicare alcune informazioni su tutti i fogli del documento stesso come ad esempio: n. di pagina, titolo del documento, nome del creatore del documento, data di produzione, oppure una semplice barra colorata e così via. Queste informazioni possono essere inserite al di sopra del testo scritto ("intestazioni") oppure al di sotto del testo scritto ("Piè di pagina") una volta per tutte.

Il comando per attivare questa funzionalità si trova alla voce **INSERISCI → Riga di intestazione /Piè di pagina**.

La stampa unione

Questa funzionalità è un comando molto importante e viene utilizzato normalmente in ambito lavorativo.
In cosa consiste:
immaginate di dover inviare a 100 dipendenti di un'azienda, una lettera nel quale si avvisa che l'azienda per motivi tecnici rimarrà chiusa un giorno lavorativo. Se non esistesse questo comando l'impiegato dovrebbe:
scrivere la lettera senza gli indirizzi; riprendersi tutti i nominativi e gli indirizzi dei 100 dipendenti; scrivere ad uno ad uno i nominativi ed indirizzi e stampare singolarmente una lettera per ogni dipendente.

Grazie al comando “Stampa Unione” invece l'impiegato, che ha già un'anagrafica dei dipendenti, deve semplicemente scrivere la lettera e, attraverso il comando Stampa unione, unirà la lettera con i dati dei 100 dipendenti e stamperà con un solo INVIO tutte le lettere.
Gli elementi importanti per poter utilizzare questo comando sono:
• la lettera tipo (il documento comune da stampare)
• un elenco di dati strutturati in forma tabellare che possono essere nominativi o indirizzi

Per realizzare questa operazione bisogna seguire i seguenti PASSI:
• creazione della lettera tipo (documento principale)
• utilizzo o creazione della tabella nella quale saranno inseriti tutti i dati
• Inserimento all'interno del documento principale di tutti i campi che conterranno le informazioni che vogliamo stampare sul documento stesso
• Unione del documento con tutti i dati contenuti nei campi

In particolare, una volta creato o aperto un documento che sarà la nostra lettera-tipo bisogna richiamare dal menu dei comandi la voce: **STRUMENTI → Stampa guidata in serie …**

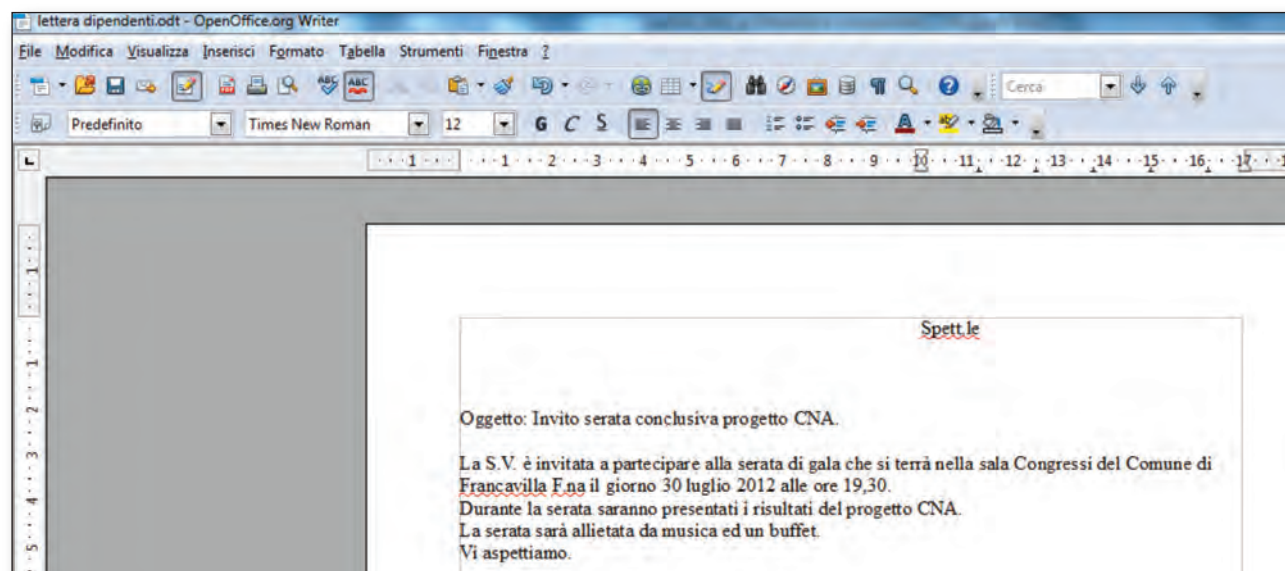
Esempio. Invito a serata di Gala
Cominciamo col creare un foglio elettronico con i dati dei dipendenti e salviamolo:

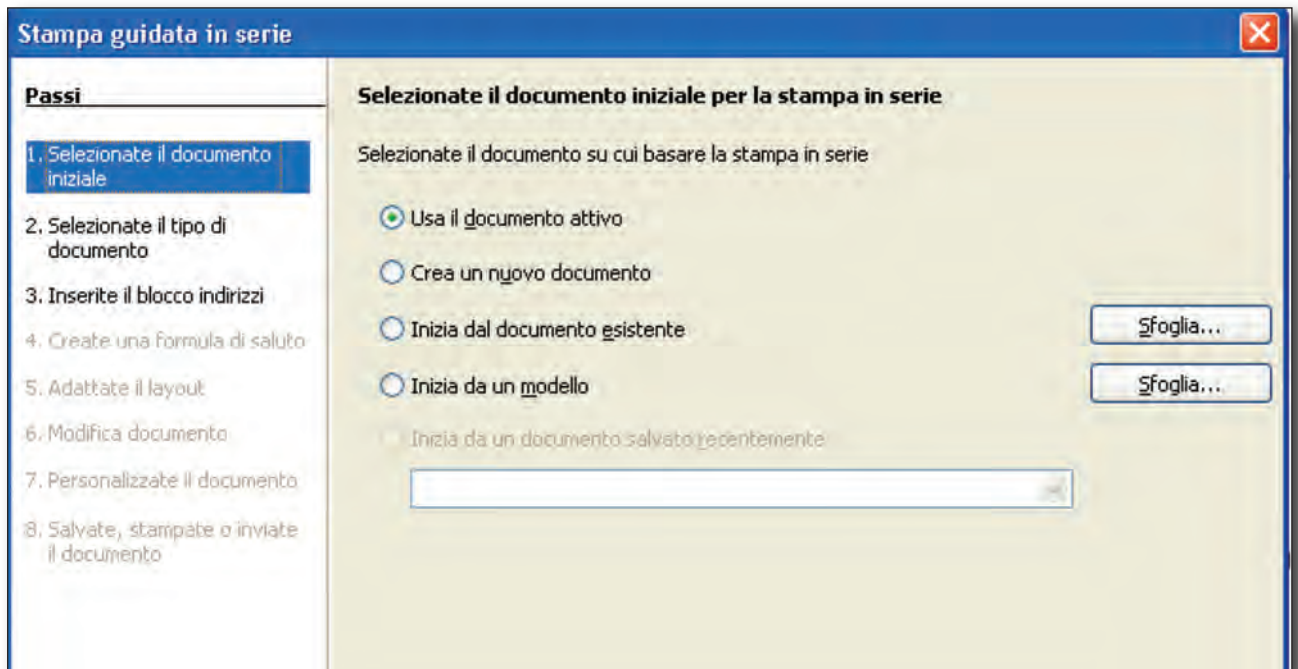
	A	B	C	D	E	F	G
1	titolo	nome	cognome	indirizzo	città	C.A.P.	
2	sig.	Giovanni	Rossi	via Pietro Micca	FRANCAVILLA FONTANA	72021	
3	sig.ra	Giorgia	Martina	via Latiano	ORIA	72024	
4							
5							
6							
7							
8							

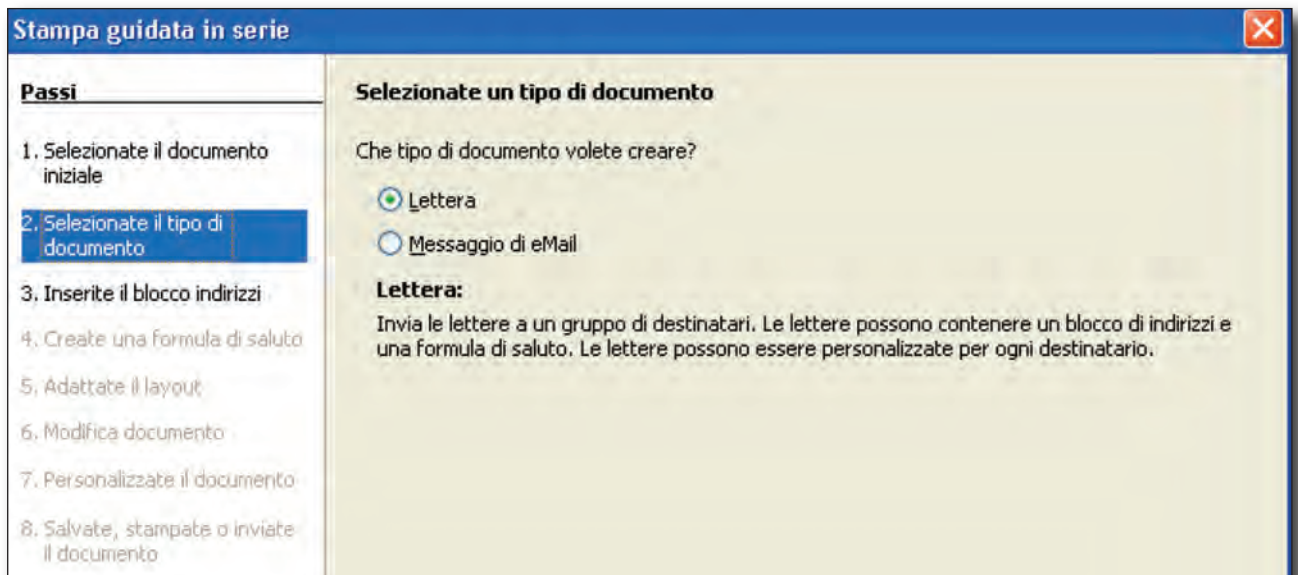
Passiamo alla creazione del documento principale

Spett.le

Oggetto: Invito serata conclusiva progetto CNA.

La S.V. è invitata a partecipare alla serata di gala che si terrà nella sala Congressi del Comune di Francavilla F.na il giorno 30 luglio 2012 alle ore 19,30.
Durante la serata saranno presentati i risultati del progetto CNA.
La serata sarà allietata da musica ed un buffet.
Vi aspettiamo.

Posizioniamoci con il cursore nel punto dove desideriamo inserire i dati dei dipendenti e richiamiamo dal menu dei comandi la voce: **Strumenti → Stampa guidata in serie …**

Verrà visualizzata la seguente videata:

Innanzitutto si sceglie l'opzione **"Usa il documento attivo"**.

Per passare da un punto all'altro bisogna utilizzare il pulsante "AVANTI". Nella nuova finestra si sceglie l'opzione "Lettera"

Si clicca sul pulsante **"Seleziona elenco di indirizzi"**.

Indicare il percorso del file dove abbiamo registrato i dati dei dipendenti e confermare:

Confermare:

Stampa guidata in serie

Passi

1. Selezionate il documento iniziale
2. Selezionate il tipo di documento
3. Inserisci il blocco indirizzo
4. Create una formula di saluto
5. Adattate il layout
6. Modifica documento
7. Personalizzate il documento
8. Salvate, stampate o inviate il documento

Inserisci il blocco indirizzo

1. Selezionate l'elenco di indirizzi contenente i dati da utilizzare. Questi dati sono necessari per la creazione del blocco di indirizzi.

Seleziona un altro elenco di indirizzi...

Elenco di indirizzi attuale: elenco dipen

2. ☑ Questo documento conterrà un blocco di indirizzi

<Titolo>
<Nome> <Cognome>
<Indirizzo, riga 1>
<C.A.P.> <Città>

<Titolo>
<Nome> <Cognome>
<Indirizzo, riga 1>
<C.A.P.> <Città>
<Paese>

Extra...

☑ Elimina righe che contengono solo campi vuoti

3. Abbinate il nome del campo usato per la stampa in serie con le intestazioni di colonna nella vostra sorgente dati.

Combina campi...

4. Controllate che i dati degli indirizzi siano corretti.

sig.
Giovanni Rossi
via Pietro Micca
72021 FRANCAVILLA FONTANA

Documento: 1

? | << Indietro | Avanti >> | Fine | Annulla

Se non interessa la formula di saluto togliere la spunta dalla voce corrispondente e confermare:

Scegliere la posizione dei dati all'interno della pagina eventualmente togliendo la spunta dalla voce **"Allinea al corpo del testo"**, modificando i parametri "Da sinistra" e "Dall'alto" e aiutandosi con l'anteprima:

Stampa guidata in serie

Passi

1. Selezionate il documento iniziale
2. Selezionate il tipo di documento
3. Inserisci il blocco indirizzo
4. Create una formula di saluto
5. Adattate il layout
6. Modifica documento
7. Personalizzate il documento
8. Salvate, stampate o inviate il documento

Adattate il layout del blocco di indirizzi e della formula di saluto

Posizione del blocco indirizzi

Allinea al corpo del testo

Da sinistra 2,50cm

Dall'alto 5,49cm

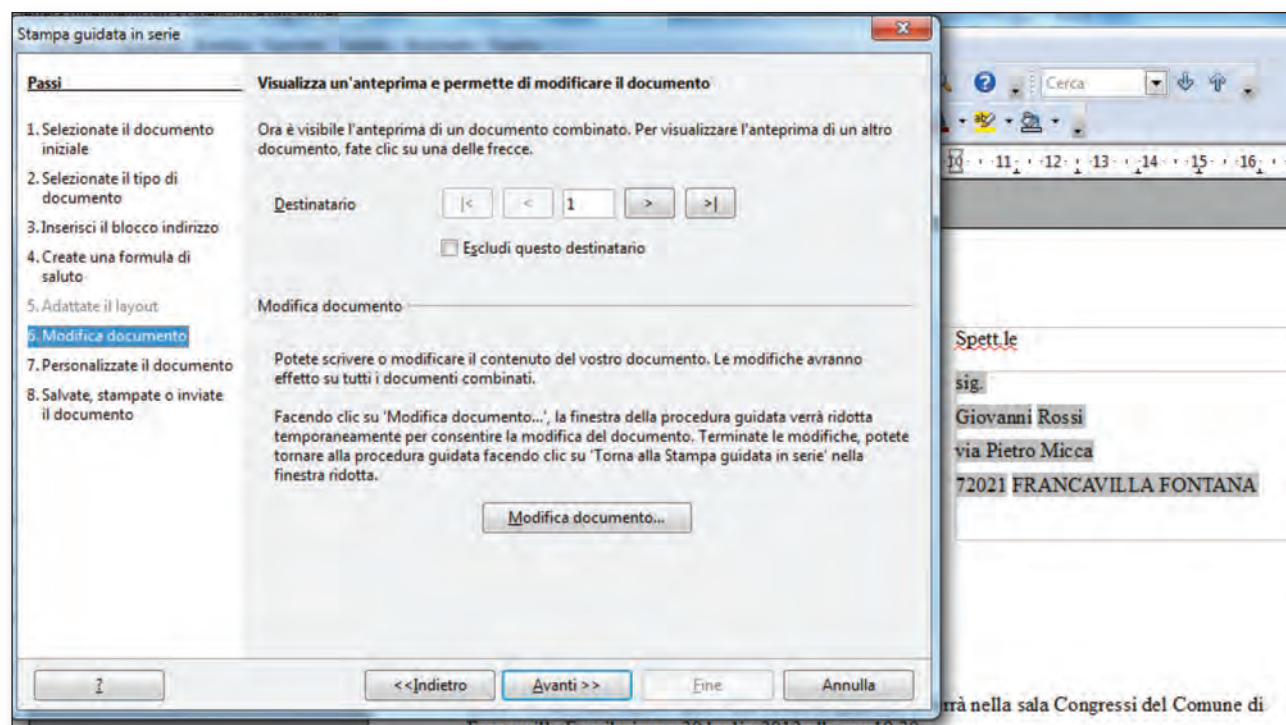
Posizione della formula di saluto

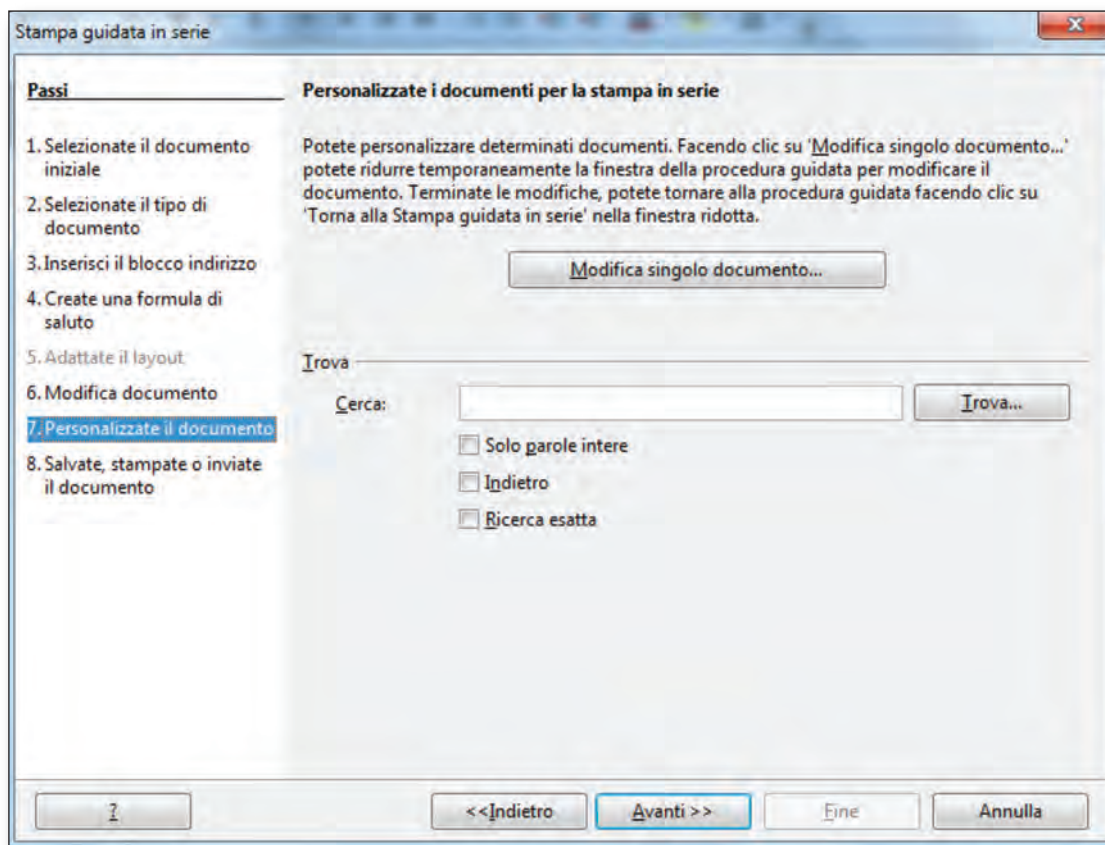
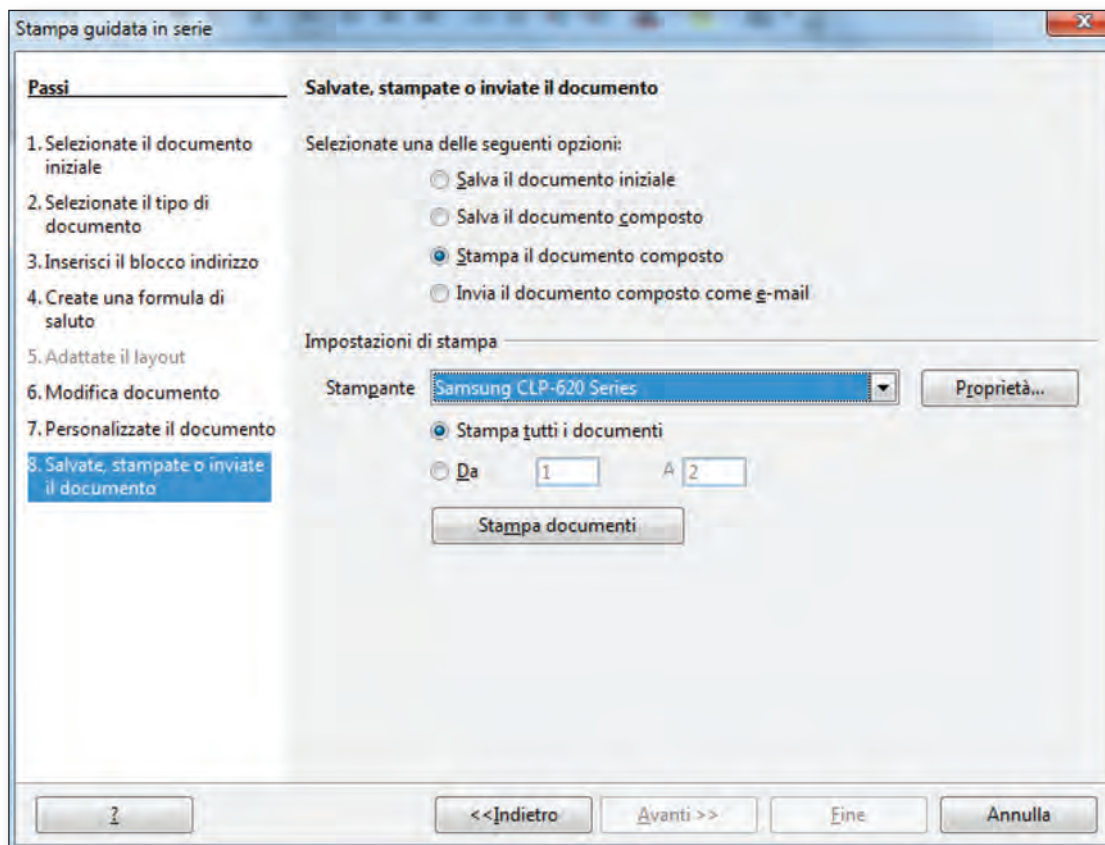
Sposta In alto

Sposta In basso

Zoom Intera pagina

? | <<Indietro | Avanti >> | Fine | Annulla

Spostare la maschera di immissione, trascinandola dalla barra del titolo, e visualizzare l'anteprima del documento:

Nell'anteprima è possibile eseguire delle modifiche e naturalmente confermarle.

A questo punto il programma permette ancora delle modifiche:

Come ultima alternativa è possibile scegliere se stampare o salvare il documento per stamparlo in seguito.

BUON LAVORO ...

Per eventuali problemi consultate la Guida in linea (help) del software.

8. Il Foglio Elettronico (OpenOffice Calc)

A cura di **Giorgia Martina** e **Salvatore Madaro**

Competenze	Abilità	Conoscenze
Utilizzare, con autonomia operativa ed organizzativa, strumenti di comunicazione visiva e multimediale, anche con riferimento alle strategie espressive e agli strumenti tecnici della comunicazione in rete.	*Utilizzare i principali software per la produttività individuale.*	*Software di utilità e software applicativi.*
	Raccogliere, organizzare e rappresentare informazioni.	*Metodi di rappresentazione dei dati e di documentazione.*
	Rappresentare ed elaborare i risultati delle misure di grandezze fisiche utilizzando strumenti informatici.	*Foglio elettronico.*

Il foglio elettronico (detto anche Foglio di Calcolo) è uno strumento utilizzato per risolvere problemi matematici di vario genere più o meno complessi.

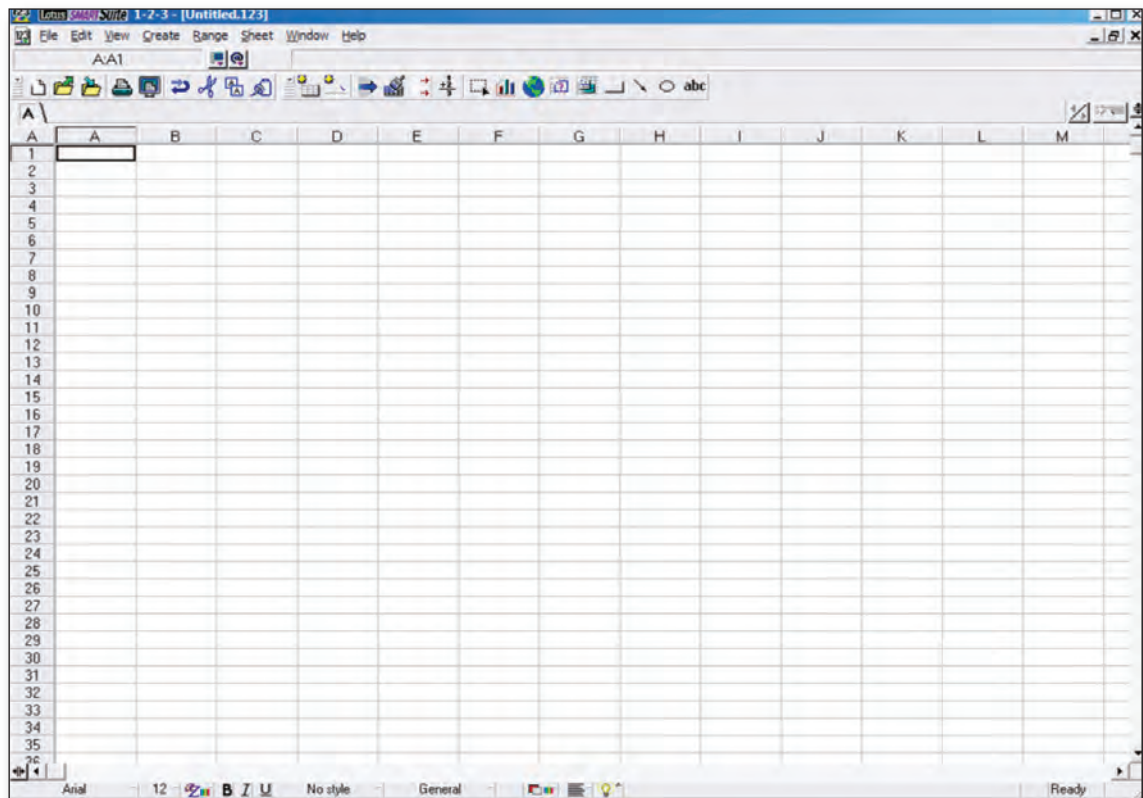
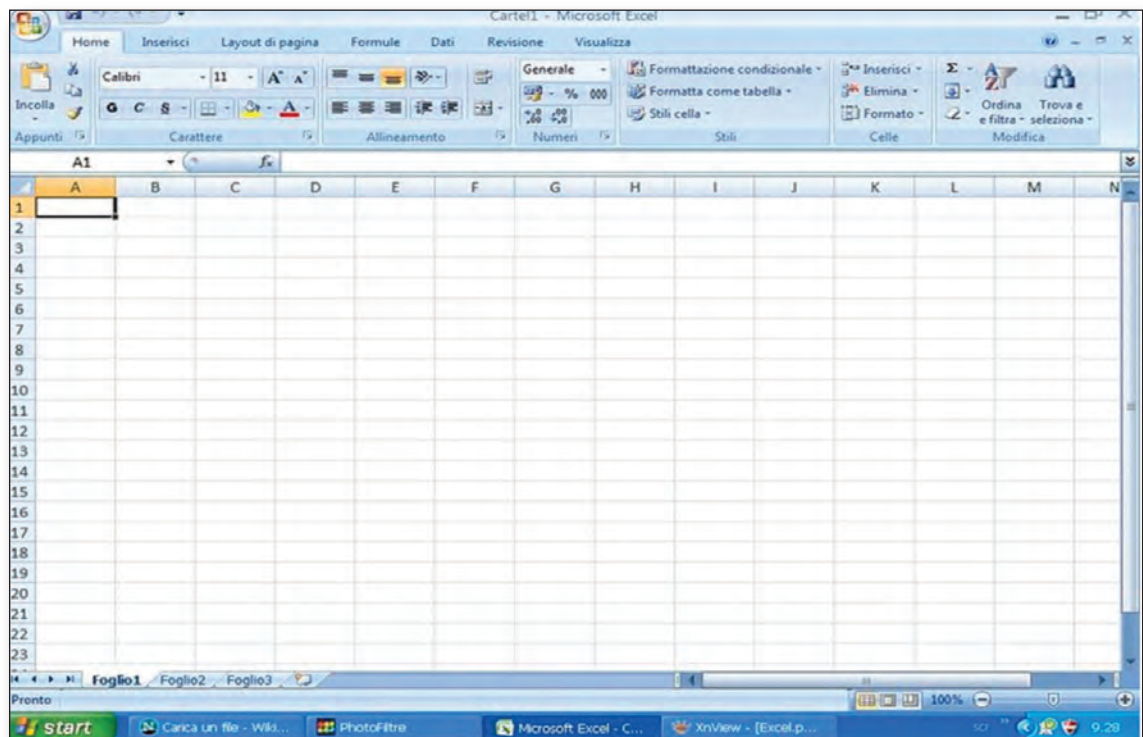
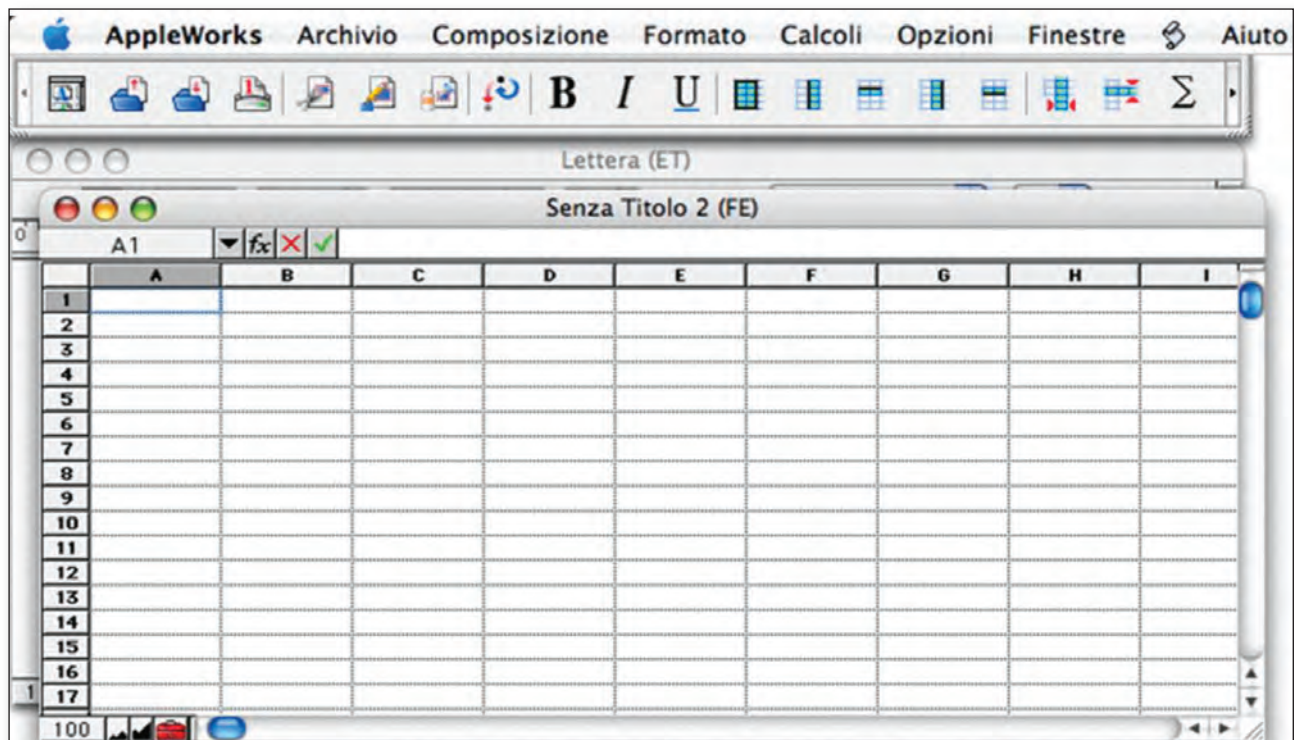
In generale questo tipo di applicativi permette di tenere sotto controllo una serie di dati numerici strutturandoli in tabelle e di eseguire calcoli semplici (come somme, prodotti, etc) oppure elaborazioni più complesse; permette, inoltre di realizzare grafici di ottima resa estetica e di semplice lettura e di elaborare informazioni non numeriche per estrarre dati da cui si ha bisogno di ottenere tabelle riassuntive.

Il foglio di calcolo elettronico è un programma interattivo, che mette a disposizione dell'utente un'area operativa strutturata in modo reticolare ed identificabile attraverso righe e colonne. Le righe sono indicate da numeri, le colonne da lettere. L'intersezione di una riga con una colonna individuano l'elemento fondamentale del foglio elettronico: LA CELLA.

Uno dei vantaggi più significativi di un foglio elettronico è il ricalcalo automatico del risultato di tutte le formule nel momento in cui viene modificato il contenuto di una cella, se questa cella è inserita, come riferimento, all'interno della formula stessa. Pertanto, una volta impostato il foglio elettronico, questo può essere usato come modello sia per l'elaborazione di nuovi insiemi di dati e sia per analisi di previsione.

Breve storia del Foglio elettronico

Partendo da **VisiCalc (Ms-Dos)**,

è nato negli anni ‘80 **Lotus 1-2-3 (S.O. Ms-Dos)**

che è stato superato da **Microsoft Office Excel (S.O. Windows).**

AppleWork INumber (S.O. Macintosh)

Al momento con l'intensificarsi delle attività volte a realizzare software libero di vario genere sono nati **OpenOffice:Calc (S.O. Windows)**

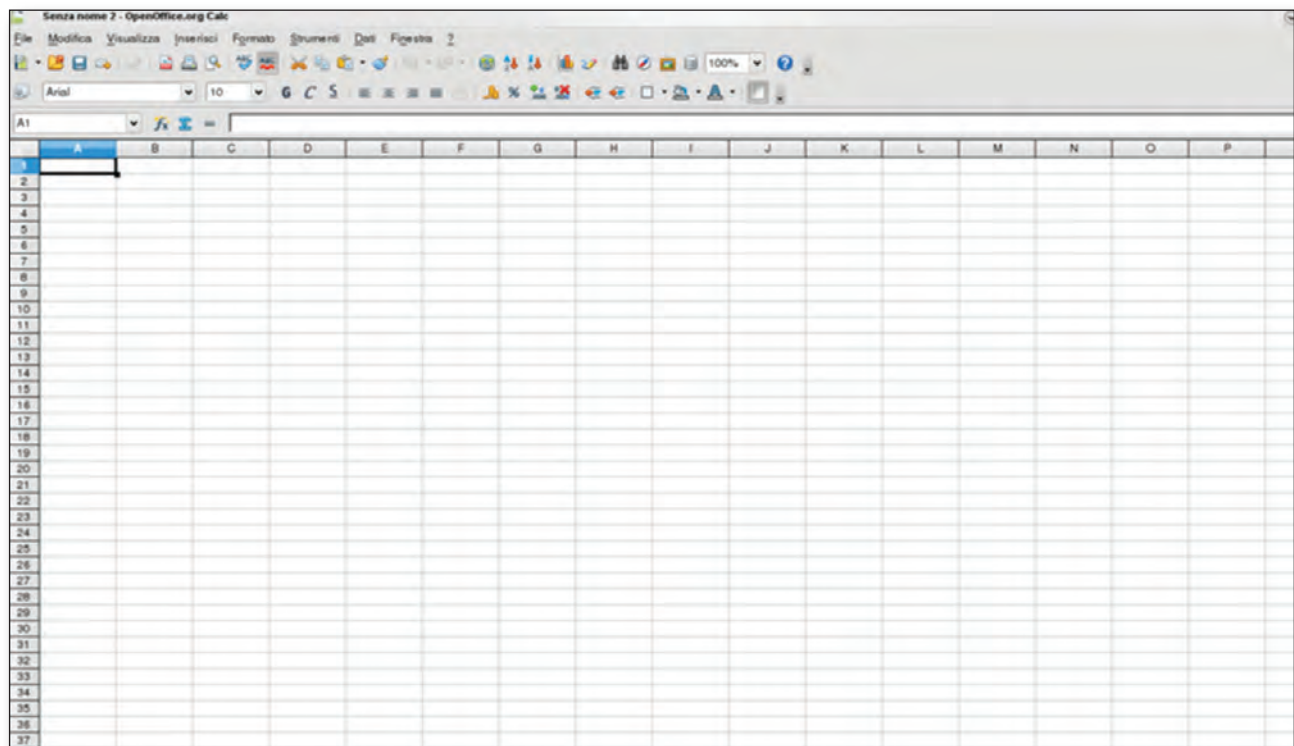
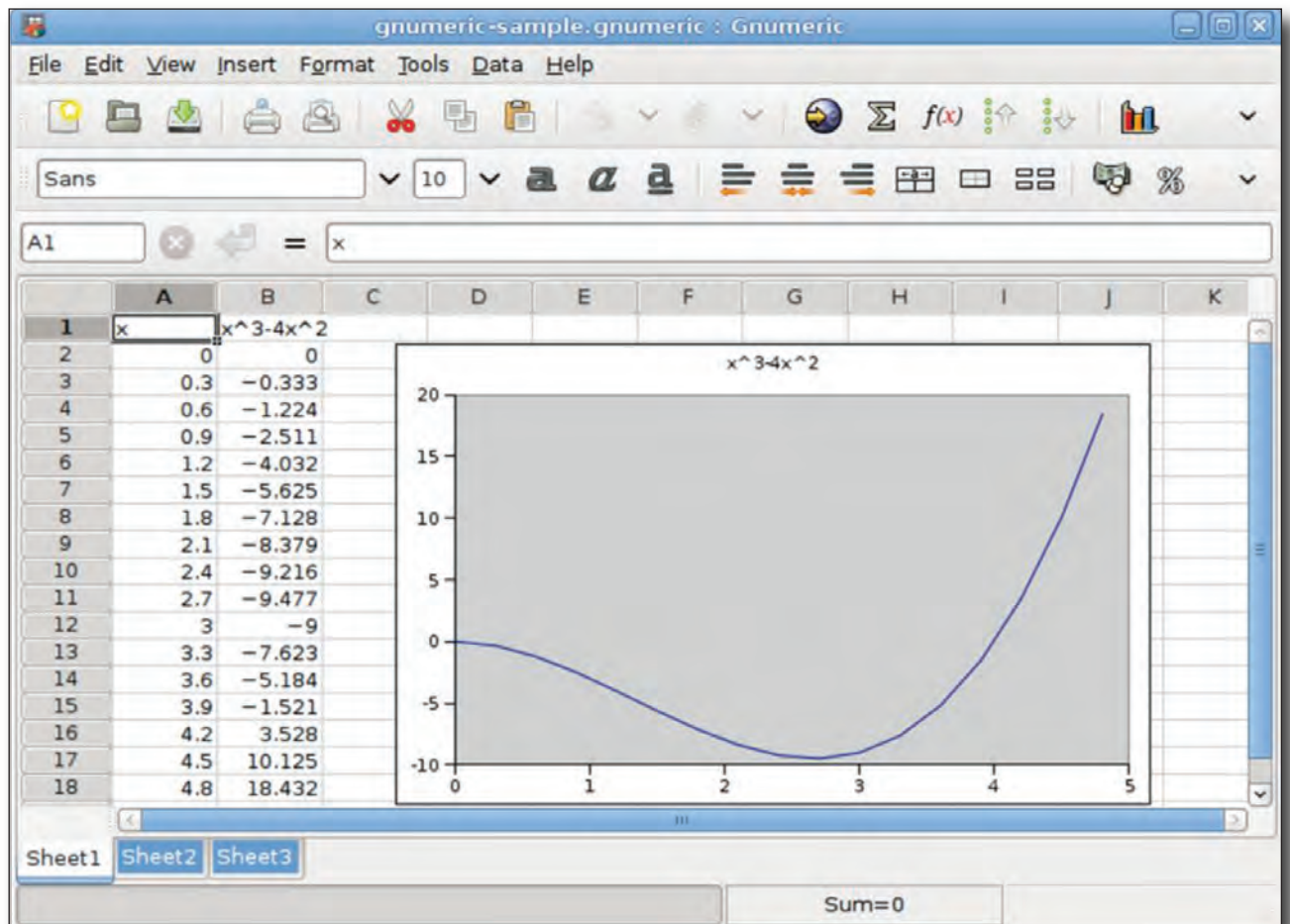
Gnumeric (S.O. Linux)

gnumeric-sample.gnumeric : Gnumeric

File Edit View Insert Format Tools Data Help

	A	B
1	x	x^3-4x^2
2	0	0
3	0.3	−0.333
4	0.6	−1.224
5	0.9	−2.511
6	1.2	−4.032
7	1.5	−5.625
8	1.8	−7.128
9	2.1	−8.379
10	2.4	−9.216
11	2.7	−9.477
12	3	−9
13	3.3	−7.623
14	3.6	−5.184
15	3.9	−1.521
16	4.2	3.528
17	4.5	10.125
18	4.8	18.432

x^3-4x^2

Sheet1 Sheet2 Sheet3 Sum=0

D'ora in avanti, in accordo con la filosofia del "Book in Progress", utilizzeremo il foglio elettronico di **Open Office CALC**, che si può scaricare gratuitamente da Internet; notate come la differenza con gli altri fogli elettronici non sia poi così tanta! Per questo i nostri problemi saranno risolti in CALC .

Per iniziare... LA CELLA

La cella è l'elemento fondamentale di un foglio elettronico, in quanto è proprio in questo contenitore che si possono inserire tutte le possibili informazioni di tipo numerico ed alfanumerico che vengono classificati in:

- ***Dati*** **(o numeri):** che comprendono le 10 cifre, i segni "+" e "-" e la virgola decimale ",".
  Normalmente è possibile fare operazioni di ogni genere con celle di tipo "DATO".

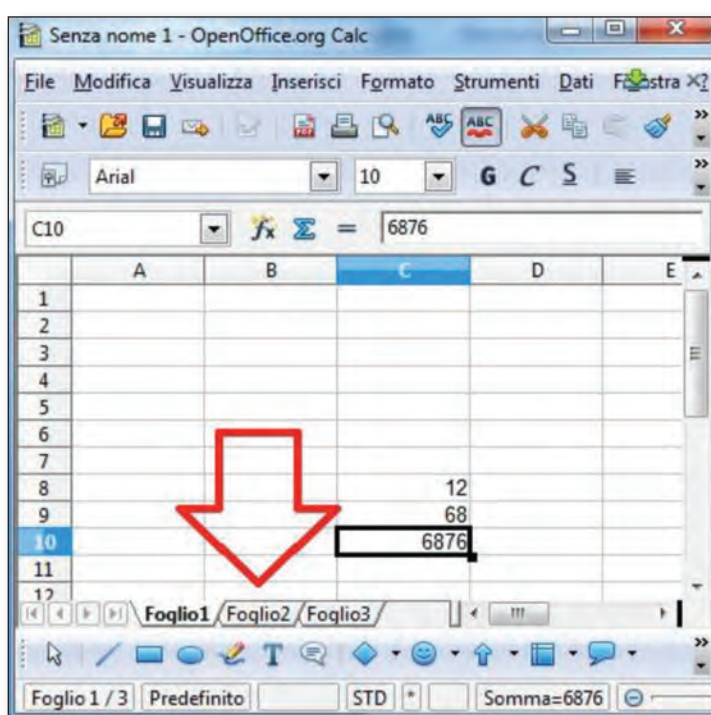
- ***Etichette*** **(o testo):** comprende tutti i numeri, lettere e simboli. Celle con questo tipo di informazione vengono utilizzate per definire un aspetto grafico di facile consultazione e comprensione.

- ***Formule*** **(espressioni matematiche):** contengono valori numerici e/o riferimenti a celle contenenti dati numerici. Il risultato dell'operazione scritta viene immediatamente visualizzata nella cella stessa. Le formule devono:
  - iniziare con il segno "**=**"
  - possono essere indicate le priorità di operazioni da eseguire con le parentesi tonde "**(**", "**)**".
  - e operazioni vengono indicate con "**+**" (addizione), "**-**" (sottrazione), "*****" (moltiplicazione) "**/**" (divisione).
  - l'eventuale elevamento a potenza viene indicato con "^" (**es** 5^2 → **5^2**)

- ***Funzioni:*** Le funzioni sono procedure memorizzate nel foglio stesso che svolgono calcoli anche complessi. Tutte le funzioni esistenti sul foglio elettronico sono suddivise in categorie (tra cui “Matematiche e trigonometriche”, “Statistica”, “Logiche”, etc).
  Una funzione viene scritta nel seguente modo:
  =NomeFunzione(argomento1;argomento2; …;argomenton) dove gli “argomenti” non sono altro che i dati in ingresso che saranno elaborati dalla funzione per ottenere un risultato in uscita.

I riferimenti di cella

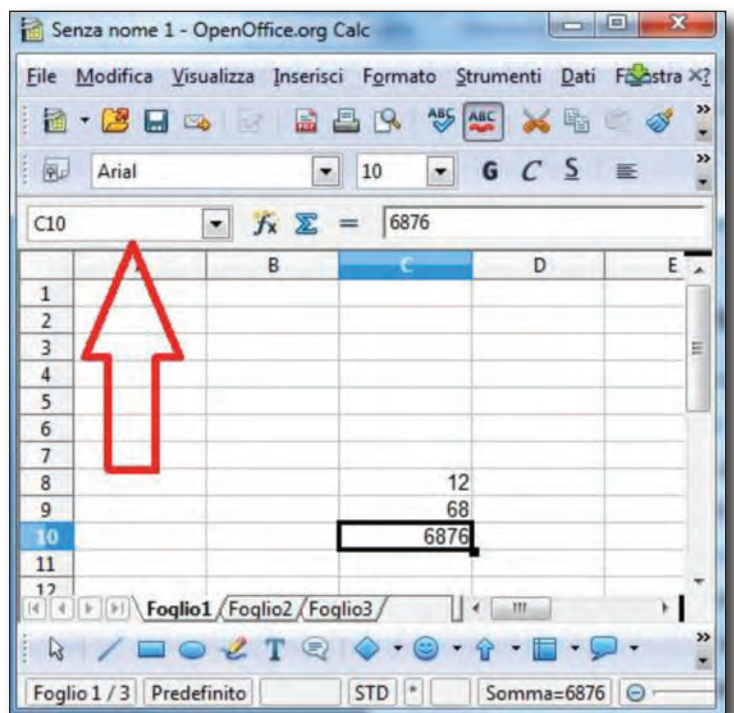
Un riferimento nel foglio elettronico rappresenta la posizione di una o più celle dell’area di lavoro e consente l’individuazione di valori o dati che si desidera utilizzare in una formula o funzione. Tramite i riferimenti è possibile utilizzare in una formula, dati contenuti in diversi punti di un foglio di lavoro oppure utilizzare il valore di una cella in più formule.
E’ anche possibile fare riferimento a celle di altri fogli della stessa cartella di lavoro (solitamente quando si apre un foglio elettronico si hanno a disposizione più fogli su cui scrivere e l’utente può decidere di aggiungere o toglierne).

È possibile copiare il contenuto di una cella in un’altra zona ma quando di copiano formule o funzioni allora l’indirizzo delle celle inserite nelle formule stesse viene aggiornato, adattandosi alla nuova posizione; in questo caso si dice che il riferimento delle celle è relativo.
Normalmente tutte le posizioni possibili che può

occupare un dato sono “riferimenti relativi”. In alcuni casi però è necessario fissare alcuni riferimenti per fare in modo che, se si copia una formula in più punti del foglio, questa non cambierà automaticamente i suoi collegamenti ad altre celle. Per fare questo è necessario inserire il simbolo “$” sia prima del riferimento di riga che di colonna se si intende bloccare entrambi gli elementi (riferimento assoluto) oppure prima di uno solo di essi:

È possibile usare ripetutamente il tasto-funzione F4 (premuto assieme al tasto delle maiuscole) per rendere assoluti i riferimenti di cella; in particolare quando si seleziona il riferimento ad una cella e si preme Maiusc +F4 si blocca contemporaneamente il riferimento a riga e colonna, premendo ancora si blocca solo il riferimento di riga e premendo di nuovo si blocca solo il riferimento di colonna. Premendo ulteriormente Maiusc+F4 si ottiene un comportamento periodico.

Operatività sul foglio

Ci si può muovere all’interno del foglio in diversi modi:

- utilizzando il mouse
- utilizzando le barre di scorrimento
- utilizzando i pulsanti PgUp e PgDw (rispettivamente Pag ↑ e Pag ↓)
- utilizzando le frecce direzionali (← ↑ → ↓) presenti nella tastiera
- scrivendo direttamente l’indirizzo della cella nella casella del nome (vedi figura a fianco).

Tutti i comandi che si decide di attivare, agiscono immediatamente nella cella attiva in quel momento (che appare con un contorno nero marcato) **e quindi è importante selezionare la o le celle interessate** utilizzando il mouse con la tecnica del “trascinamento” per celle adiacenti oppure tenendo premuto il tasto “CTRL” e cliccando su celle non adiacenti.

Per cancellare i dati inseriti precedentemente, invece, si può:
- selezionare la cella o la zona e premere il tasto “**Canc**”
- selezionare la cella o la zona e andare alla voce di menu “**Modifica” → Elimina contenuti**”

Per copiare o spostare il contenuto di una cella o più celle, utilizzare i comandi **Copia/Incolla** (per duplicare le informazioni presenti nelle celle), **Taglia/Incolla** (per spostare le informazioni in un’altra parte del foglio).

Problemi relativi alle formule e ai valori di errori

Se non è possibile calcolare il risultato di una formula, verrà visualizzato un valore di errore.
E' possibile ad esempio che si verifichi un errore se si utilizza del testo laddove la formula richiede un valore numerico, se si elimina una cella a cui fa riferimento una formula oppure se si utilizza una cella non sufficientemente ampia da consentire la visualizzazione del risultato. Per esempio, tra gli indicatori di messaggi di errori più comuni troviamo:

Errore #####:	*Il valore numerico immesso in una cella è troppo esteso per essere visualizzato all'interno della cella. E' possibile ridimensionare la colonna trascinando il bordo che si trova tra le intestazioni di colonna.*
#VALORE!	*Viene visualizzato quando viene utilizzato un tipo di argomento o di operando errato, e cioè un'immissione di testo quando la formula richiede un numero. Controllare che le celle contengano valori validi per l'operazione che si vuole fare.*
#DIV/0!	*Viene visualizzato quando il divisore di una formula è uguale a 0.*

Elenco delle funzioni più usate in un foglio elettronico

=somma()	*Restituisce la somma di un gruppo di celle*
=media()	*Calcola la media aritmetica di un gruppo di celle*
=min()	*Visualizza il valore più piccolo in un gruppo di celle.*
=max()	*Visualizza il valore più grande in un gruppo di celle*
=potenza()	*Restituisce il risultato di un numero elevato a potenza*
=radq()	*Restituisce la radice quadrata di un numero*

E così via ….

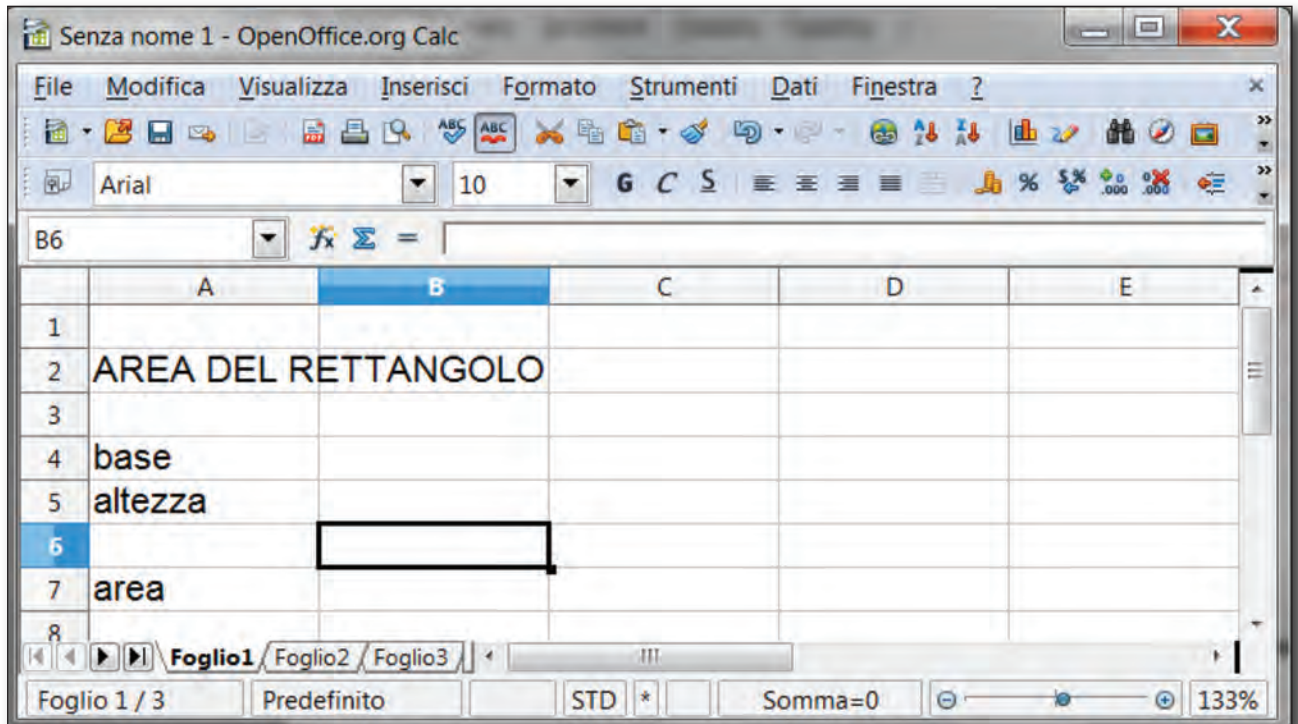
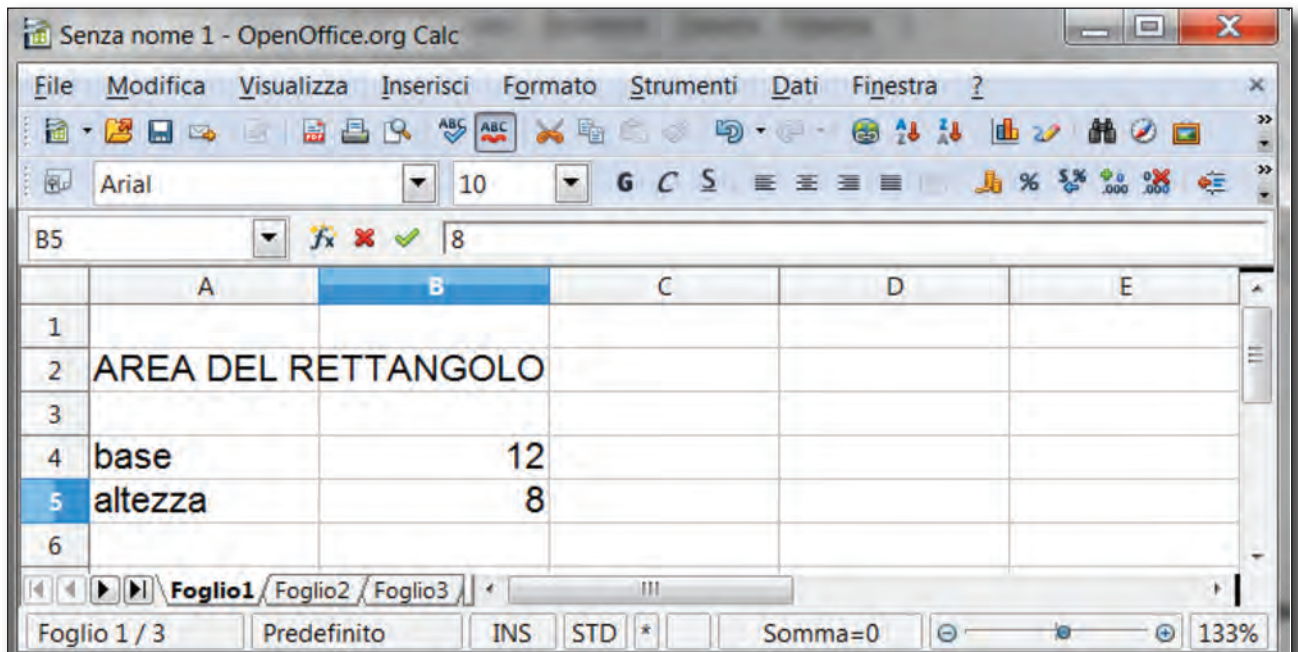
Tutte le funzioni utilizzabili con i fogli elettronici si possono trovare alla voce "**Inserisci" → "Lista delle funzioni**". In questo riquadro di dialogo potrai trovare una funzione e con la creazione guidata potrai fare tutti i calcoli che desideri.
In alternativa vicino alla "**Riga di digitazione**" è presente il pulsante per la "**Creazione guidata formula**".

Osserviamo che quando all'interno di una cella scriviamo una formula o una funzione, quando confermiamo con il tasto "**Invio**" noi leggiamo nella cella solo il risultato delle diverse operazioni mentre continuiamo a leggere la formula o la funzione nella "**Riga di digitazione**"

Esercizi

Esercizio n.1

Aree di rettangoli

Inizieremo a risolvere una classe di problemi molto semplici: il calcolo dell'area dei rettangoli.
Prepariamo il foglio scrivendo gli elementi essenziali del problema:

Ora inseriamo i valori della base e dell'altezza

Infine nella cella B7 scriviamo la formula che leghi tra loro i valori dei dati inseriti e mostri il valore dell'area; ogni formula inizia con il simbolo di uguale ("=") e poi bisogna dichiarare dove sono i dati e come bisogna elaborarli: attenzione, si scrive l'indirizzo dove sono contenuti i valori e non i valori stessi. Scriviamo " =B4*B5" (senza le virgolette) per indicare che si deve

moltiplicare il contenuto della cella B4 con quella della cella B5; confermiamo con “Invio” e notiamo che nella cella B7 è presente il valore dell’area mentre nella Riga di digitazione è visualizzata la formula.

Esercizio n.2

Quadrati, cubi e radici quadrate dei primi 10 numeri interi positivi

Cominciamo ad inserire le etichette quindi, usando gli appositi strumenti, modifichiamo il tipo e la dimensione del carattere e applichiamo l’effetto “Grassetto” dopo aver selezionato le celle interessate:

Sotto la cella A2 scriviamo i primi due numeri interi positivi e successivamente trasciniamo verso il basso il pulsantino di riempimento automatico; apparirà così la successione dei numeri sui quali dobbiamo eseguire i calcoli.

	A	B	C
1			
2	numero	quadrato	cubo
3	1		
4	2		
5	3		
6	4		
7	5		
8	6		
9	7		
10	8		
11	9		
12	10		
13			

	A	B	C	D	E
1					
2	numero	quadrato	cubo	radice quadrata	
3	1	=A3^2	=A3^3	=RADQ(A3)	
4	2				
5	3				
6	4				
7	5				
8	6				
9	7				
10	8				
11	9				
12	10				

Ora nelle celle B3, C3, D3, bisogna inserire le corrispondenti formule per calcolare il quadrato, il cubo e la radice quadrata del contenuto della cella A3 ossia “=A3^2”, “=A3^3” e “=radq(A3)”. (Naturalmente le virgolette non vanno scritte).

Man mano che scriviamo le formule, nelle celle appariranno i risultati mentre, selezionando le singole celle, nella riga di digitazione potremo vedere le formule inserite.

Utilizzando come prima il pulsante di riempimento automatico ricopiamo le formule nelle celle sottostanti ed il gioco è fatto.

	A	B	C	D	E
1					
2	numero	quadrato	cubo	radice quadrata	
3	1	1	1	1	
4	2	4	8	1,41	
5	3	9	27	1,73	
6	4	16	64	2	
7	5	25	125	2,24	
8	6	36	216	2,45	
9	7	49	343	2,65	
10	8	64	512	2,83	
11	9	81	729	3	
12	10	100	1000	3,16	

	A	B	C	D
1				
2	numero	quadrato	cubo	radice quadrata
3	1	1	1	1
4	2	4	8	1,41
5	3	9	27	1,73
6	4	16	64	2
7	5	25	125	2,24
8	6	36	216	2,45
9	7	49	343	2,65
10	8	64	512	2,83
11	9	81	729	3
12	10	100	1000	3,16

Magari possiamo dare un aspetto migliore al foglio se selezioniamo le celle che contengono i valori e applichiamo un allineamento al centro.

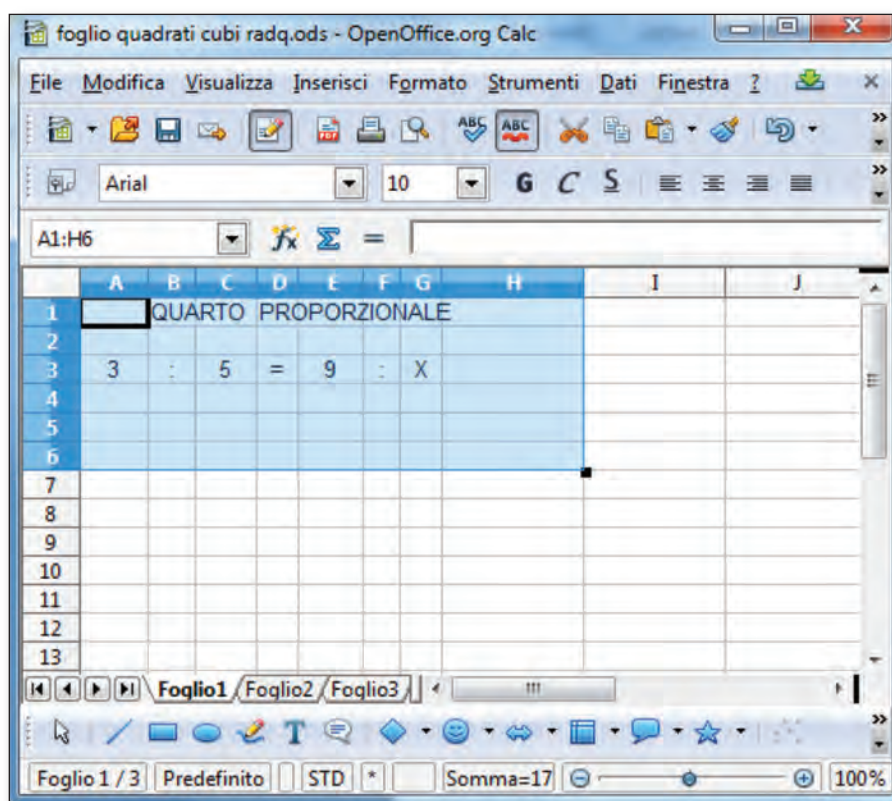
Osserviamo che è stato necessario allargare la colonna D per far entrare nella cella D2 tutta l’etichetta; tale risultato si può ottenere tenendo premuto il tasto sinistro del mouse mentre il cursore è nel punto di separazione tra le intestazioni delle colonne D ed E ed allargando la colonna D oppure semplicemente facendo un doppio clic mentre il cursore è sulla linea di separazione dell’intestazione delle colonne.

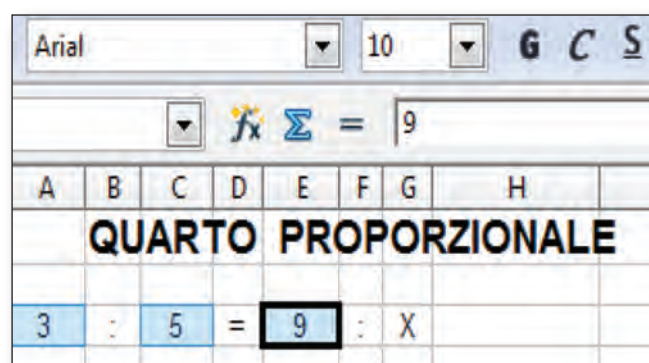
	A	B	C	D
1				
2	numero	quadrato	cubo	radice quadrata
3	1	1	1	1
4	2	4	8	1,41
5	3	9	27	1,73
6	4	16	64	2
7	5	25	125	2,24
8	6	36	216	2,45
9	7	49	343	2,65
10	8	64	512	2,83
11	9	81	729	3
12	10	100	1000	3,16

Esercizio n.3

Risolvere le proporzioni

Ora intendiamo creare un foglio per la risoluzione delle proporzioni; terremo conto dei diversi casi che si possono presentare e, per ognuno di essi scriveremo la formula risolutiva.

Intanto scriviamo le etichette ed inseriamo alcuni valori; riduciamo la larghezza delle colonne (vedi esercizio n. 2). Dobbiamo fare attenzione a scrivere in celle separate i valori numerici distinguendoli dai simboli alfanumerici . Selezioniamo le celle con i valori e le etichette e applichiamo un allineamento centrale.

Osserviamo che per scrivere il carattere "=" è stato necessario premettere un apice per far riconoscere al foglio elettronico che non è l'inizio di una formula.

Aumentiamo la dimensione del carattere e applichiamo il carattere grassetto al titolo.

Ora vogliamo evidenziare le celle nelle quali si potranno inserire valori numerici; per fare questo è necessario selezionare delle celle che non sono vicine (non contigue) tenendo premuto il tasto "Ctrl" mentre si clicca sulle diverse celle e quindi si sceglie il colore di sfondo con il tasto: ed il colore del carattere con il tasto ottenendo un effetto più gradevole.

	A	B	C	D	E	F	G	H
1		QUARTO PROPORZIONALE						
2								
3	3	:	5	=	9	:	X	
4								
5								

Questo serve anche per mostrare ad un eventuale utente del foglio elettronico quali sono le celle sulle quali intervenire.

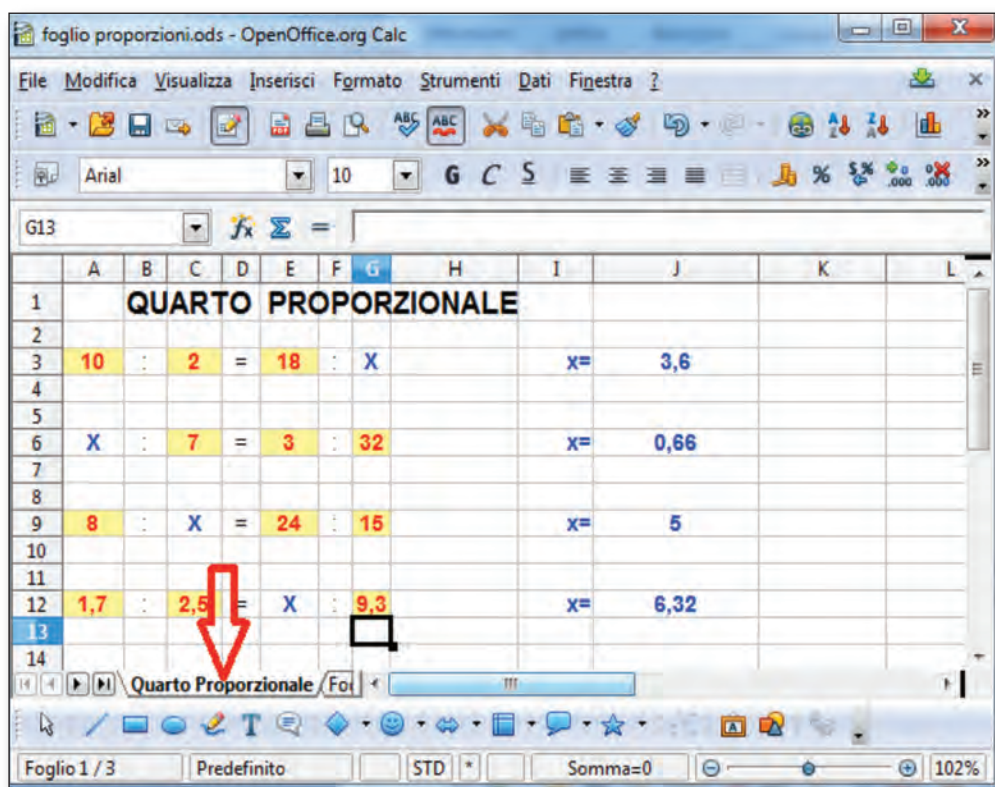
Inseriamo la formula che risolve la proporzione: "=C3*E3/A3".

Analogamente si procede quando l'incognita x si trova nell'altro estremo oppure tra i medi.

	A	B	C	D	E	F	G	H	I	J
1		QUARTO PROPORZIONALE								
2										
3	3	:	5	=	9	:	X		x=	=C3*E3/A3
4										
5										
6	X	:	5	=	9	:	15		x=	=C6*E6/G6
7										
8										
9	3	:	X	=	9	:	15		x=	=A9*G9/E9
10										
11										
12	3	:	5	=	X	:	15		x=	=A12*G12/C12
13										

Alla fine variando i valori numerici otterremo nuove soluzioni per le proporzioni.

	A	B	C	D	E	F	G	H	I	J
1		QUARTO PROPORZIONALE								
2										
3	10	:	2	=	18	:	X		x=	3,6
4										
5										
6	X	:	7	=	3	:	32		x=	0,66
7										
8										
9	8	:	X	=	24	:	15		x=	5
10										
11										
12	1,7	:	2,5	=	X	:	9,3		x=	6,32

Rinominiamo questo foglio di lavoro “Quarto Proporzionale “ con un clic destro sulla linguetta in basso per distinguerlo dagli altri fogli nei quali risolviamo il problema del medio proporzionale e del terzo proporzionale.

Ecco il calcolo del medio proporzionale e del terzo proporzionale:

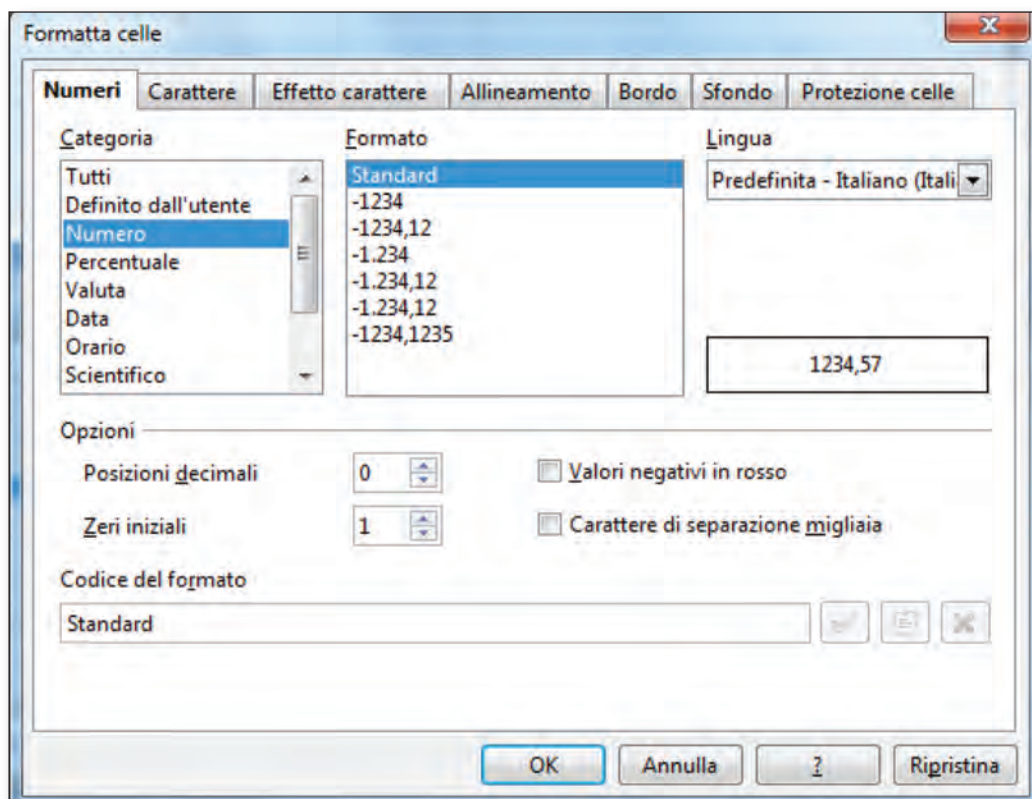
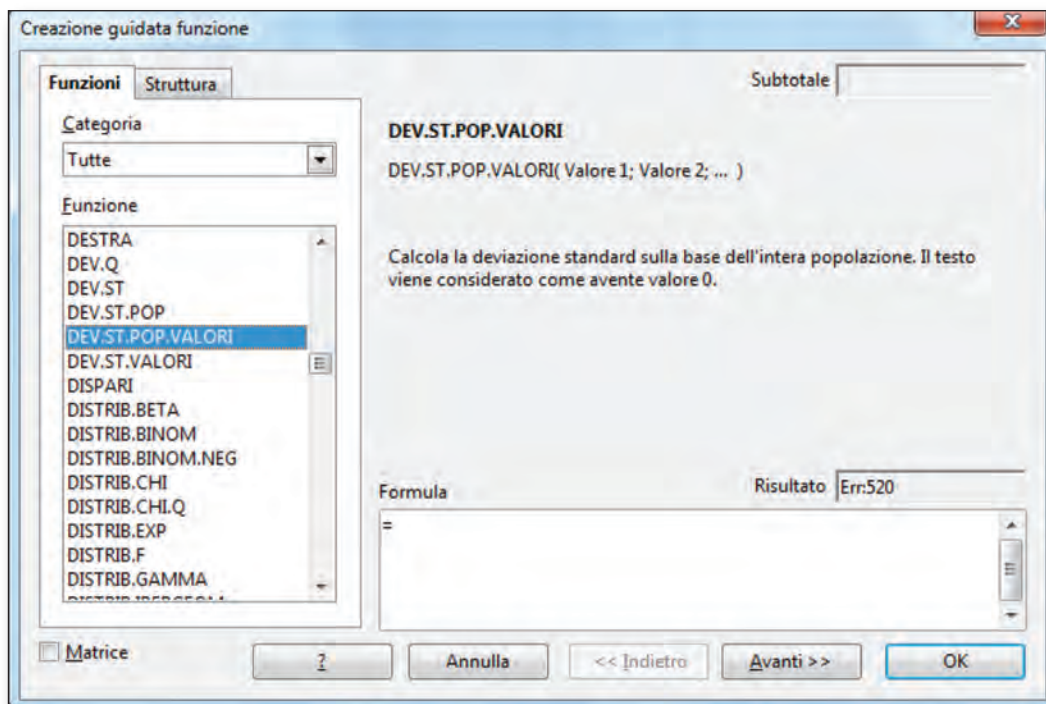
Anche questo foglio è stato rinominato; per passare da un foglio all'altro basta cliccare sulle relative linguette. Se gli altri fogli non sono visibili basta agire ai pulsanti di avanzamento:

Esercizio n.4

Semplici Calcoli Statistici

Nel laboratorio di Fisica sono state effettuate le seguenti rilevazioni sperimentali della massa di un oggetto e si intende ricavare i principali valori statistici (media aritmetica, valore minimo, valore massimo, errore assoluto, errore percentuale, scarti, deviazione standard). Ricordiamo che la media aritmetica si calcola eseguendo la somma di tutti i valori e dividendo per il loro numero; il foglio elettronico però offre una funzione apposita.

L'errore assoluto commesso in una serie di misure è dato dalla semidifferenza tra il valore massimo ed il valore minimo; l'errore relativo è il rapporto tra l'errore assoluto e la media aritmetica mentre l'errore percentuale si ricava moltiplicando l'errore relativo per 100. Di seguito le principali formule applicate e i valori corrispondenti:

Osserviamo che le celle da B8 e B9 hanno un numero di cifre decimali maggiore per visualizzare dei valori molto piccoli. Per ottenere questo si agisce nel menu dei comandi:
Formato → celle → si modifica il numero di decimali da visualizzare. Dalla stessa posizione si può intervenire per altre personalizzazioni della cella; si consiglia di provare le diverse opportunità. Osserviamo che alcune funzioni che abbiamo introdotto (Media, Min , Max) agiscono su un intervallo di celle e questo viene rappresentato scrivendo gli indirizzi delle due celle estreme separati da " : ". Nel caso dell'errore percentuale, nella corrispondente formula non è stato eseguito il prodotto per 100 ma è stato assegnato alla cella il formato percentuale utilizzando lo stesso percorso descritto sopra.

Adesso inseriamo gli scarti e la deviazione standard. Per quest'ultima funzione usiamo il tasto di creazione guidata funzione ; si apre una scheda con un elenco molto consistente di funzioni di diversa natura con la relativa descrizione.

La situazione finale si presenta come segue:

	A	B	C	D	E	F	G	H
1		VALORI STATISTICI FONDAMENTALI						
2	RILEVAZIONI							
3	6,14	6,13	6,15	6,18	6,12	6,16	6,17	6,15
4								
5	media aritm=	=MEDIA(A3:H3)						
6	minimo=	=MIN(A3:H3)						
7	massimo=	=MAX(A3:H3)						
8	errore assoluto	=(B7-B6)/2						
9	errore relativo	=B8/B5						
10	Errore %	=B9						
11								
12	SCARTI							
13	=A3-$B$5	=B3-$B$5	=C3-$B$5	=D3-$B$5	=E3-$B$5	=F3-$B$5	=G3-$B$5	=H3-$B$5
14								
15	deviaz stand	=DEV.ST.POP(A3:H3)						

Con i valori numerici:

	A	B	C	D	E	F	G	H
1		VALORI STATISTICI FONDAMENTALI						
2	RILEVAZIONI							
3	6,14	6,13	6,15	6,18	6,12	6,16	6,17	6,15
4								
5	media aritm=	6,15						
6	minimo=	6,12						
7	massimo=	6,18						
8	errore assoluto	0,0300						
9	errore relativo	0,0049						
10	Errore %	0,49%						
11								
12	SCARTI							
13	-0,01	-0,02	0	0,03	-0,03	0,01	0,02	0
14								
15	deviaz stand	0,02						

Osserviamo che nel calcolo degli scarti la formula è stata scritta solo nella cella A13 utilizzando il riferimento relativo per A3 ed il riferimento assoluto per B5 in modo che una volta ricopiata la formula, la parte con riferimento relativo si aggiorna automaticamente e quella con riferimento assoluto rimane sempre uguale.

Esercizio n.5

Grafico delle temperature

In laboratorio si è portato ad ebollizione un liquido registrando le seguenti temperature al variare del tempo.
Si vuole realizzare un grafico che rappresenti il fenomeno.
Occorre selezionare i dati dell'intervallo di celle A4:B12 quindi si preme il pulsante .

Si attiva così la "procedura guidata grafico" nella quale occorre selezionare le voci che interessano per ottenere il grafico che più si presta a rappresentare il fenomeno.

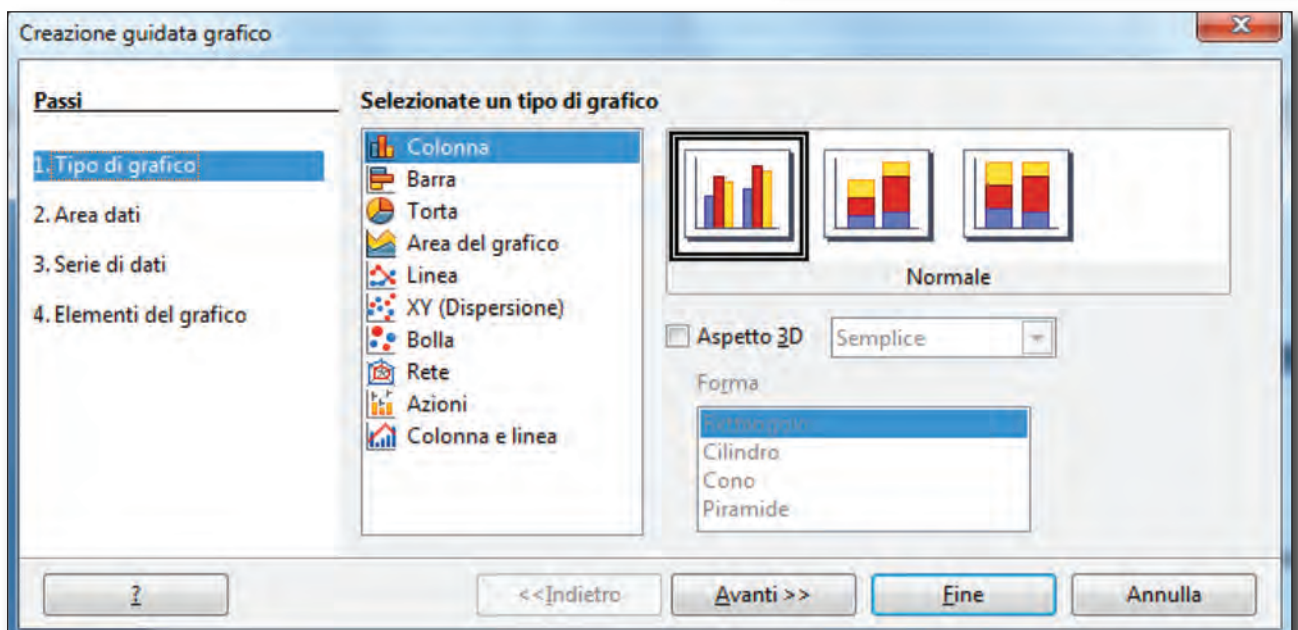
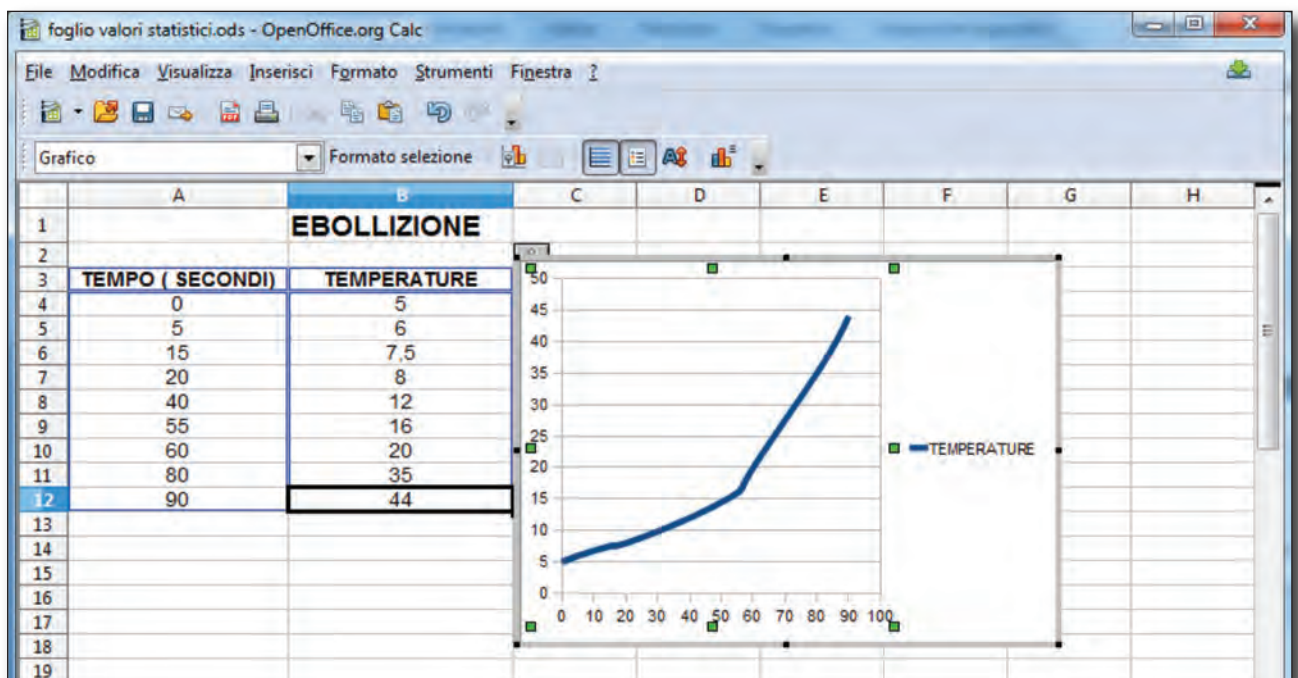
	A	B
1		EBOLLIZIONE
2		
3	TEMPO (SECONDI)	TEMPERATURE
4	0	5
5	5	6
6	15	7,5
7	20	8
8	40	12
9	55	16
10	60	20
11	80	35
12	90	44
13		

Nel nostro caso selezioniamo il tipo di grafico xy con l'effetto seguente:

Esercizio n.6

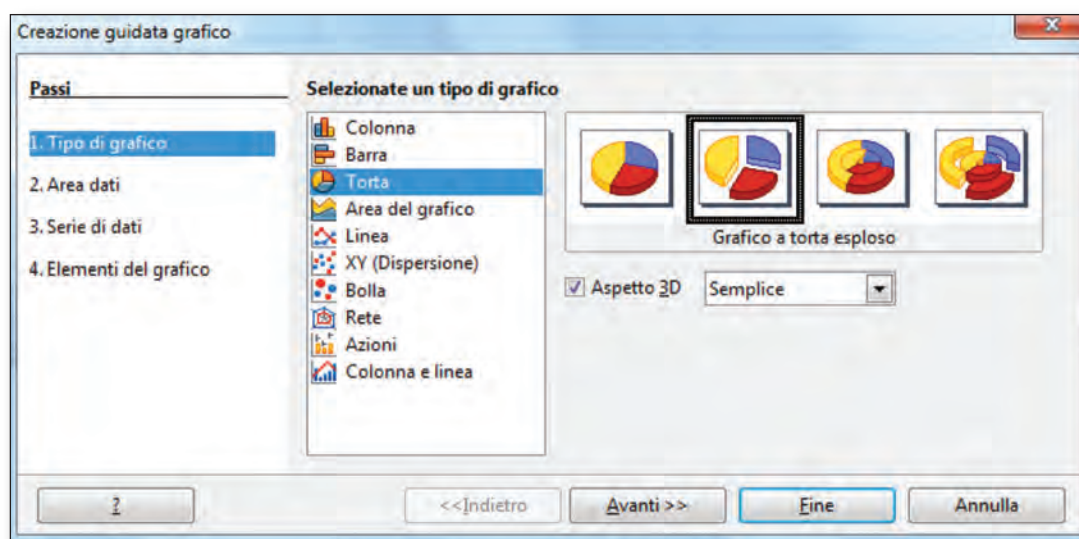
Grafico delle componenti del personale della scuola

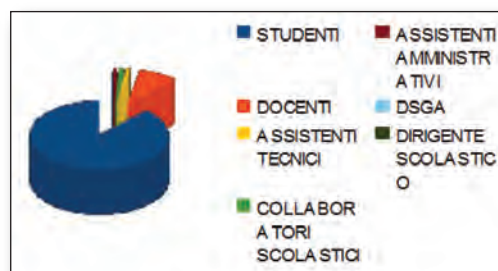
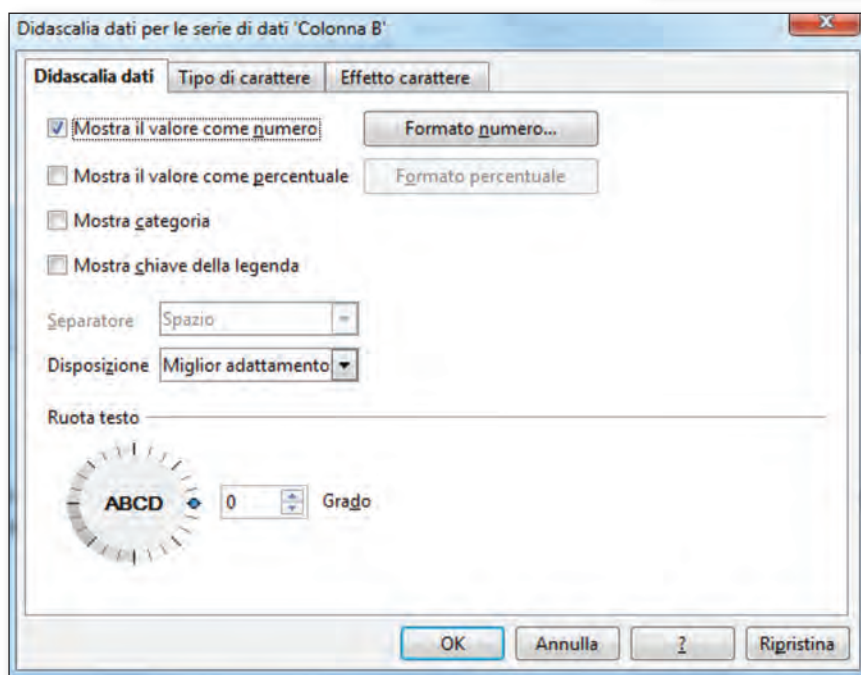
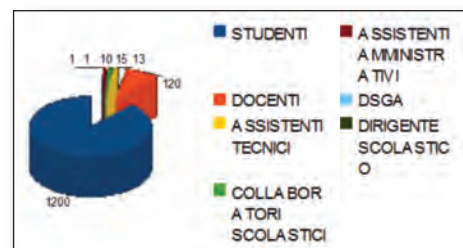
Si vuole rappresentare con un grafico a torta le componenti di un istituto scolastico.

Innanzitutto inseriamo le etichette e formattiamole con dei caratteri di dimensione 14 punti, grassetto e di colore blu; adeguiamo la dimensione delle colonne in modo che contengano per intero le etichette ed applichiamo l'allineamento centrale. Osserviamo che per selezionare una intera colonna oppure una intera riga è sufficiente cliccare sulla relativa intestazione mentre per selezionare l'intero foglio basta cliccare sul tassellino all'incrocio tre le intestazioni di riga e le intestazioni di colonna. Introduciamo i valori e assegniamo un colore rosso su fondo giallo.

	A	B	C
1	COMPONENTI DELLA SCUOLA		
2			
3			
4	STUDENTI	1200	
5	DOCENTI	120	
6	ASSISTENTI TECNICI	13	
7	COLLABORATORI SCOLASTICI	15	
8	ASSISTENTI AMMINISTRATIVI	10	
9	DSGA	1	
10	DIRIGENTE SCOLASTICO	1	
11			

Ora selezioniamo i dati e clicchiamo sul tasto di creazione guidata grafico .

Scegliamo un grafico a torta 3D esploso

Personalizziamo il grafico nella voce "Elementi del grafico" ed il gioco è fatto:

Facciamo un clic destro sul grafico appena creato e selezioniamo la voce "Inserisci etichette dati" per visualizzare sul grafico anche i valori numerici inseriti.

Poiché ci interessa vedere le percentuali delle singole componenti facciamo ancora un clic desto sul grafico a torta e selezioniamo la voce “Formato etichette dati” e nella nuova scheda scegliamo “Mostra il valore come percentuale” e deselezioniamo “Mostra il valore come numero”.

Ecco l’effetto finale:

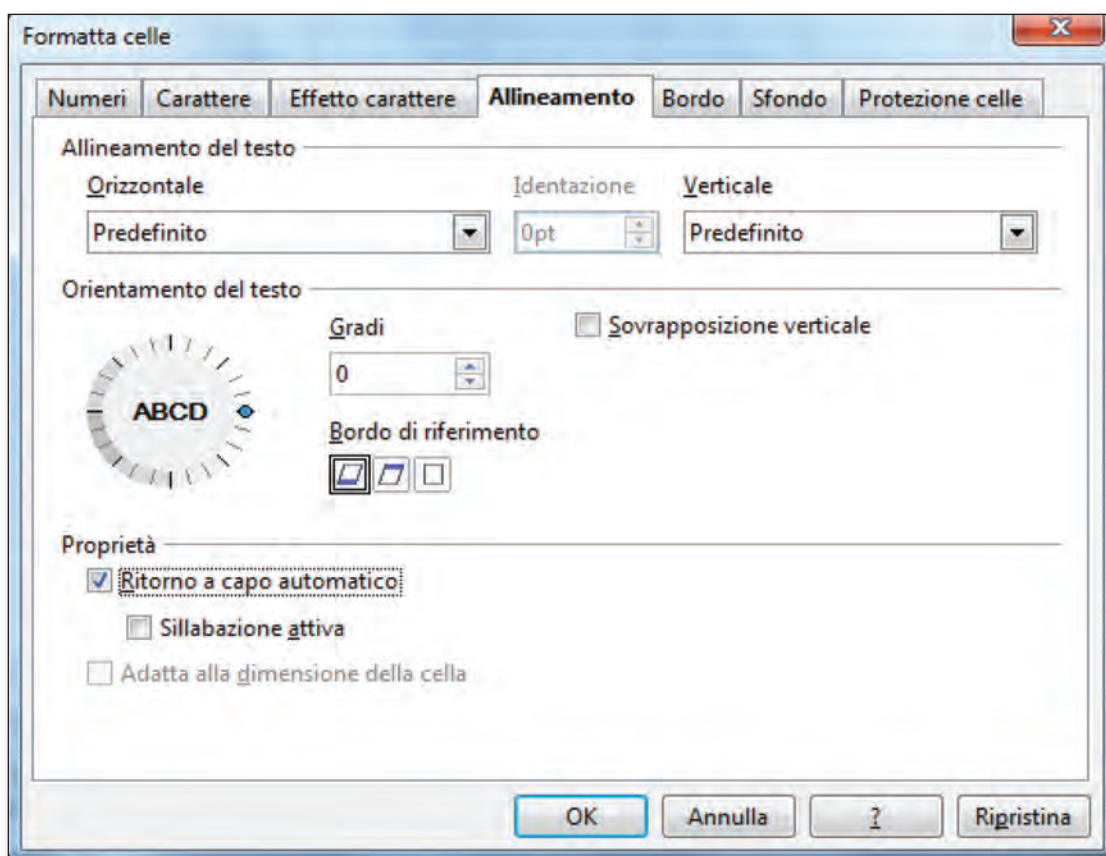
Esercizio n.7

Il conto personale

Intendiamo progettare un semplice foglio di calcolo che ci aiuti a registrare gli introiti e le spese, che evidenzi con colori diversi le spese e gli incassi e mostri costantemente la disponibilità finanziaria residua.

Cominciamo con l'inserimento delle etichette e dei primi valori:

	A	B	C	D
1		CONTO		
2				
3	DATA	SALDO INIZIALE	SOMMA	SALDO FINALE
4	01/01/10			2000
5	02/01/10	2000	150	2150
6				

Osserviamo che il testo presente nella cella B3 ha un ritorno a capo e quello della cella D3 ha un orientamento obliquo; questi effetti si possono ottenere dalla barra dei menu seguendo il percorso "Formato" →"Celle" → scegliere la cartella "Allineamento" nella nuova finestra che si apre:

Scriviamo nel foglio le formule iniziali e poi le ricopiamo in basso con il pulsantino di riempimento automatico.

	A	B	C	D
1		CONTO		
2				
3	DATA	SALDO INIZIALE	SOMMA	SALDO FINALE
4	01/01/10			2000
5	02/01/10	=D4	150	=B5+C5
6				

Inseriamo quindi gli incassi e le uscite. Ora faremo rappresentare al foglio elettronico le entrate con colore blu e leuscite con colore rosso ricorrendo alla formattazione condizionata mediante il percorso "Formato" → "Formattazione condizionata" e scegliendo le voci adeguate dalla apposita scheda.

Ecco il risultato finale:

	A	B	C	D	E	F
1		CONTO				
2						
3		saldo conto=		355		
4						
5		Conto in attivo				
6	DATA	SALDO INIZIALE	SOMMA	SALDO FINALE		
7	01/01/10			500	Inizio	
8	02/01/10	500	150	650	Versamento	
9	05/01/10	650	-75	575	Pagamento tasse scolastiche	
10	10/01/10	575	200	775	Regalo della nonna	
11	31/01/10	775	-300	475	Riparazione scooter	
12	04/02/10	475	-50	425	Pizza con gli amici	
13	08/02/10	425	80	505	Lavoro con lo zio	
14	15/02/10	505	-150	355	Tuta nuova	
15						

Il valore della cella D3 si ottiene con la formula **=SOMMA(C8:C104;D7)** che addiziona al valore iniziale presente nel conto **(D7)** tutte le successive operazioni **(C8:C104)**.
Nella cella **B5** è stata inserita la formula **=SE(D3>0**; "**Conto in attivo**";"**Conto scoperto**") che utilizza la funzione “ **=se**” per scrivere “Conto in attivo” quanto il saldo è maggiore di zero (la condizione è scritta come D3>0) e “Conto scoperto” in caso contrario.

Esercizio n.8

Densità di una sostanza

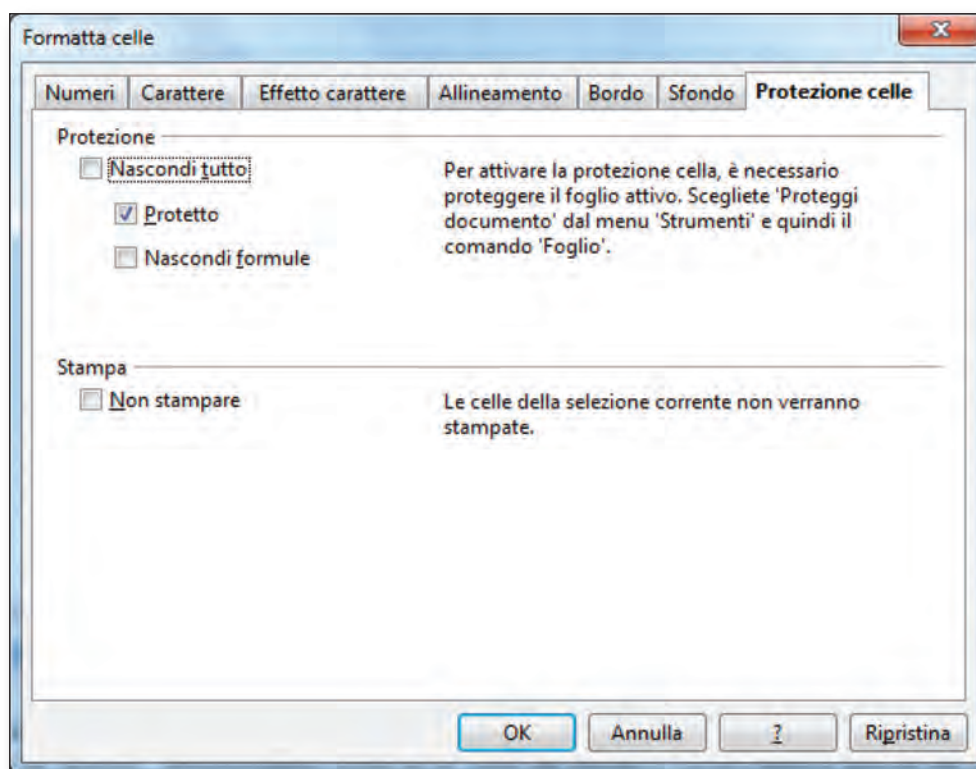
Si intende calcolare la densità di una sostanza e verificare che essa è indipendente dalla quantità di sostanza considerata. A tal proposito faremo delle misurazioni della massa e del volume su quattro corpi della stessa sostanza e riportiamo nel foglio di calcolo i dati come in figura:

	A	B	C	D
1				
2		DENSITA'		
3				
4		Massa (g)	Volume (cm³)	Densità (g/cm3)
5	corpo A	15,4	1,63	
6	corpo B	21,12	2,25	
7	corpo C	28,72	3,06	
8	corpo D	35,11	3,74	

Ora nella cella D5 inseriamo la formula per il calcolo della densità =B5/C5 e ricopiamo in basso: i riferimenti relativi si adatteranno alla nuova posizione. Evidenziamo le celle nelle quali sono previste le immissioni.

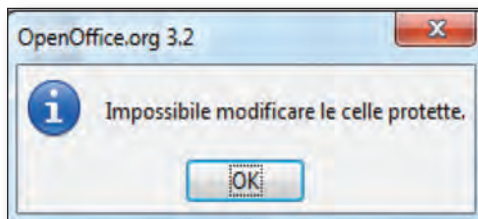
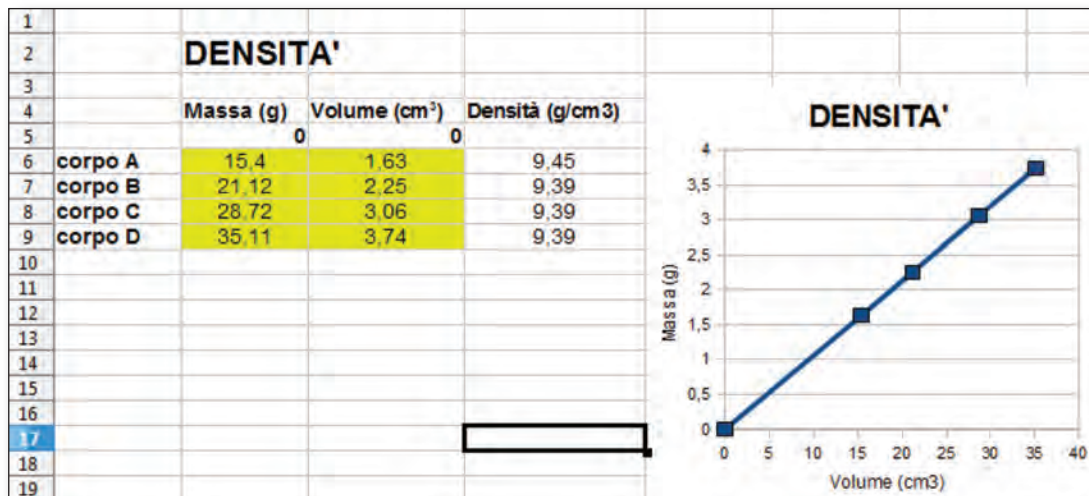
	A	B	C	D
1				
2		DENSITA'		
3				
4		Massa (g)	Volume (cm³)	Densità (g/cm3)
5	corpo A	15,4	1,63	=B5/C5
6	corpo B	21,12	2,25	=B6/C6
7	corpo C	28,72	3,06	=B7/C7
8	corpo D	35,11	3,74	=B8/C8

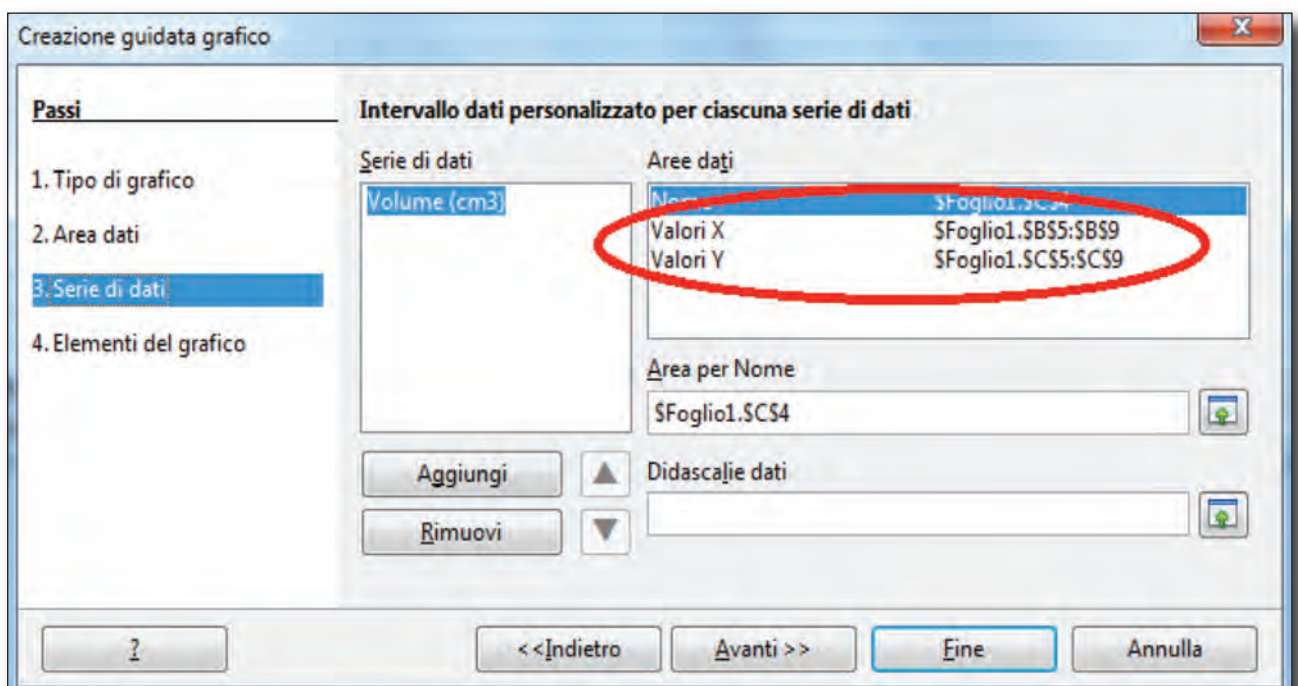
Blocchiamo le celle con le formule in modo che non vi sia qualche modifica accidentale.

Per far questo, selezioniamo soltanto le celle da rendere modificabili e quindi dal menu seguiamo il percorso: **Formato → Celle → scheda "Protezione celle"** ed eliminiamo il segno di spunta dalla voce "**Protetto**" e confermiamo con il tasto "**OK**".

Passiamo di alla voce " Strumenti" del menu e scegliamo "Proteggi documento". Da qui è possibile indicare anche una password per evitare modifiche da parte di altre persone.

Rappresentiamo il grafico volume- massa con i dati inseriti:

DENSITA'

	Massa (g)	Volume (cm³)	Densità (g/cm3)
	0	0	
corpo A	15,4	1,63	9,45
corpo B	21,12	2,25	9,39
corpo C	28,72	3,06	9,39
corpo D	35,11	3,74	9,39

Se tentiamo di scrivere in una cella protetta si presenterà il messaggio di errore.

Dobbiamo osservare il grafico che si ottiene è quello di una proporzionalità diretta tra massa e volume; la pendenza della retta rappresentata nel grafico indica la densità della sostanza.
Precisiamo che è stata inserita la coppia di valori (0,0) (massa=0 quando volume=0) per fare in modo che la retta del grafico inizi dall'origine degli assi inoltre il grafico presenta sull'asse x i volumi e sull'asse y le masse nonostante nel foglio la colonna delle masse preceda quella dei volumi e quindi automaticamente il foglio elettronico disporrebbe le masse sull'asse x. Per forzare il foglio di calcolo a fare al contrario bisogna modificare gli indirizzi della serie dei dati durante la creazione guidata del grafico:

Basta selezionare la serie e assegnare i nuovi indirizzi nella riga "Area nome".

Un modo più semplice è quello di cambiare l'ordine delle colonne del volume e delle masse.

Esercizio n.9

Soluzioni e concentrazioni

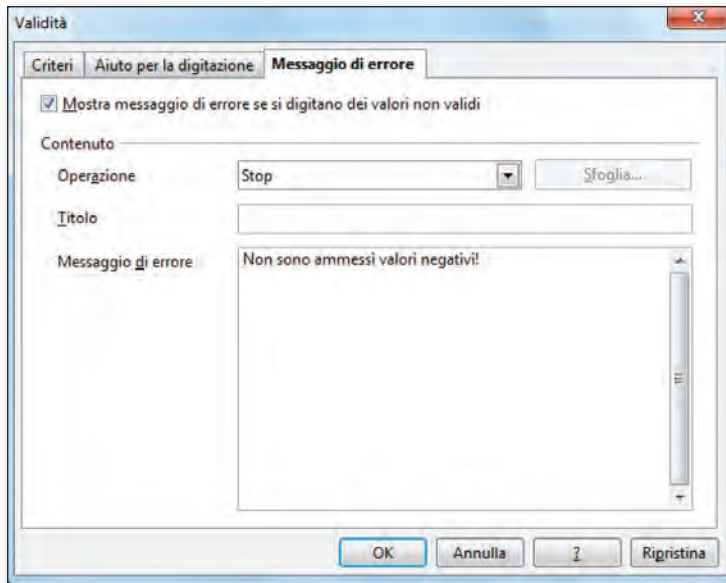
Il foglio di calcolo offre un aiuto per stabilire la concentrazione delle soluzioni.

Per stabilire la concentrazione di una soluzione occorre sapere quanto soluto è presente in 100 unità di soluzione. Solitamente la concentrazione è espressa come la massa di soluto presente in 100 g di soluzione oppure come il volume di soluto presente in 100 cm³. Nel foglio non inseriremo le unità di misura. Se indichiamo con St la quantità di soluto, Sz la quantità di soluzione e con c la concentrazione, vale seguente proporzione:

St : Sz = C : 100 e quindi $C = \frac{St}{Sz} * 100$, $St = \frac{Sz}{100} * C$, $Sz = \frac{St}{C} * 100$.

Il seguente foglio è predisposto in modo che dati due elementi dell'insieme {St, Sz,c}, si possa determinare l'elemento mancante. Inoltre se indichiamo con Sv la quantità di solvente, vale la relazione Sz = St + Sv e quindi è previsto anche il calcolo del solvente Sv con la formula Sv = Sz – St.

	A	B	C	D	E	F	G	H	I
2		SOLUZIONI							
4	INSERIRE I DATI NELLE CELLE EVIDENZIATE								
7	SOLUTO	SOLUZIONE	SOLVENTE	CONCENTRAZIONE		SOLUTO	SOLUZIONE	SOLVENTE	CONCENTRAZIONE
8	20	400	=B8-A8	=A8/B8		5	=F8/I8	=G8-F8	10,00%
20	SOLUTO	SOLUZIONE	SOLVENTE	CONCENTRAZIONE					
21	=B21*D21	300	=B21-A21	30,00%					

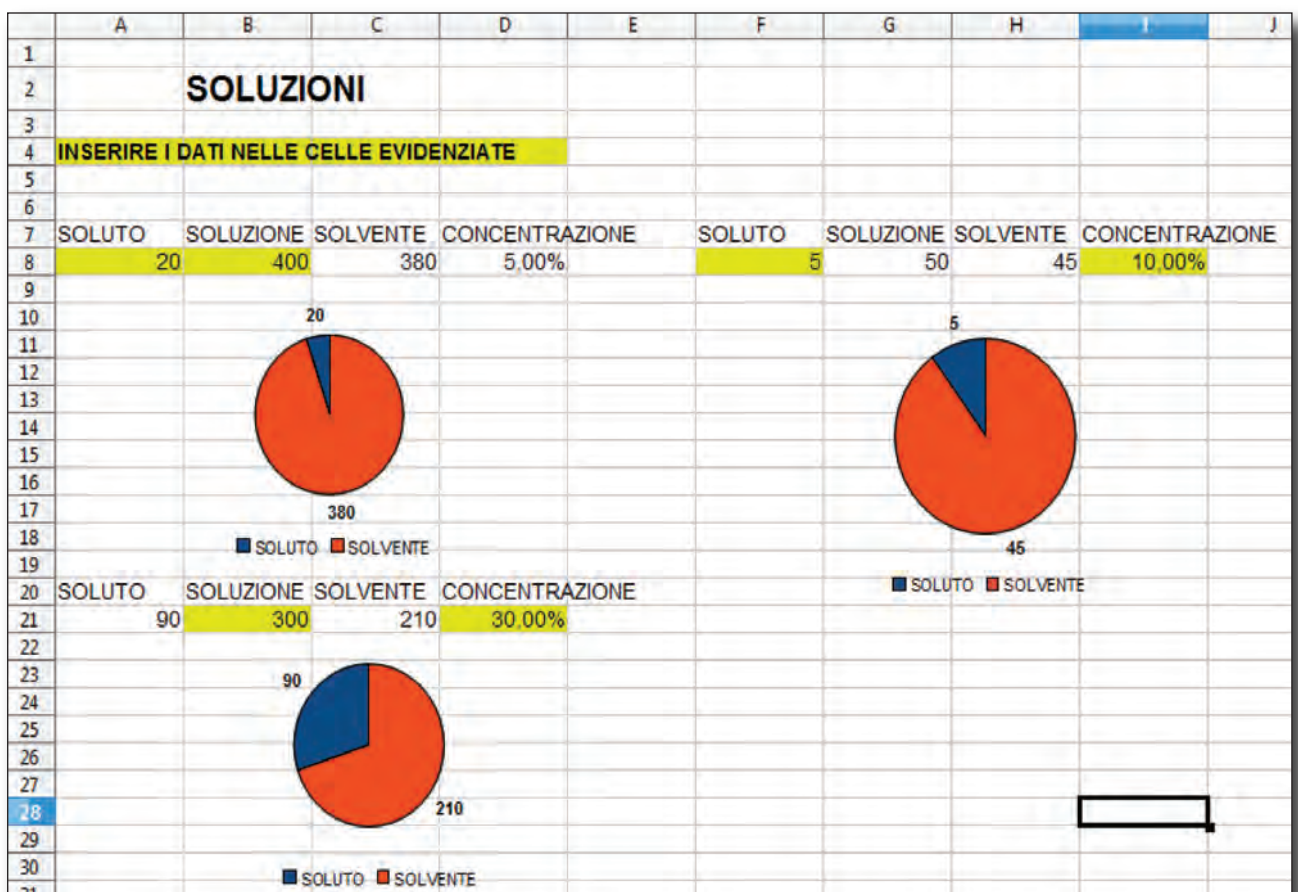
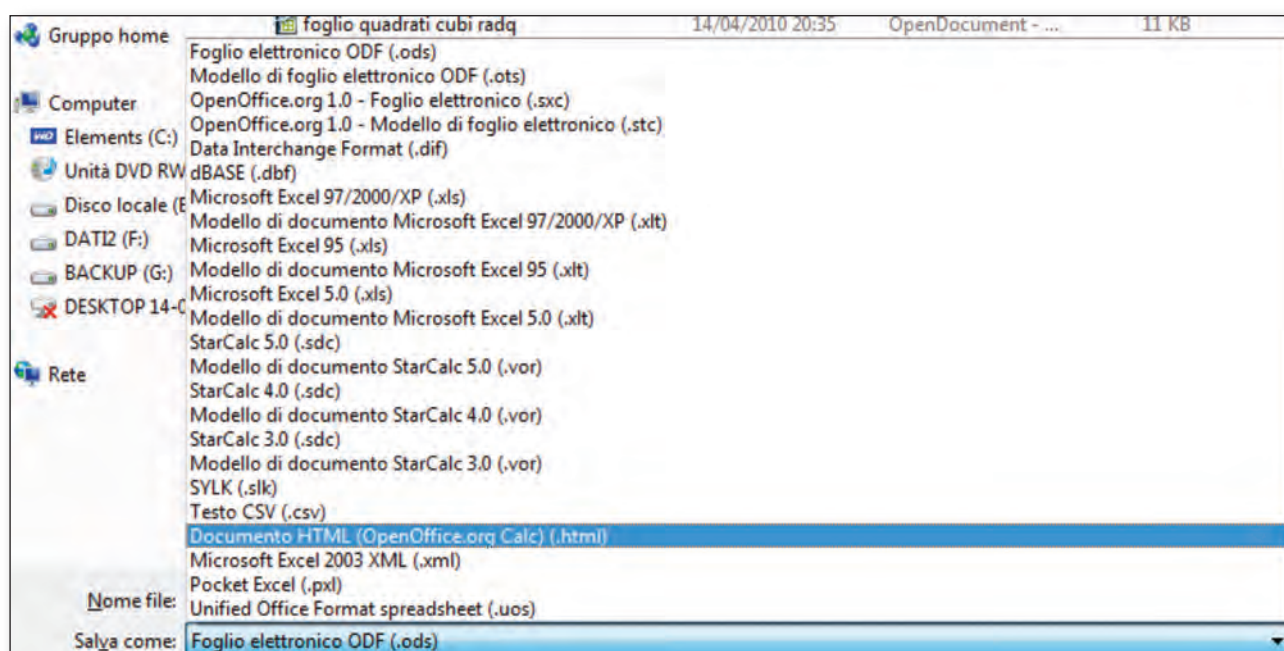
Le celle corrispondenti alla concentrazione hanno il formato percentuale (Formato → Celle → Numeri → Percentuale); nel foglio sono sbloccate solo le celle evidenziate per evitare che accidentalmente si cancellino o si modifichino le celle con le formule (Si scelgono le celle del foglio che si intende sbloccare quindi si passa a Formato → Celle → Protezione celle → si toglie la spunta dalla voce Protetto; poi si passa a Strumenti → Proteggi documento). Se vogliamo che abbiano senso i valori immessi nel foglio è necessario richiedere che questi siano non negativi e per fare questo impostiamo una regola di validazione seguendo il percorso Dati → Validità e quindi impostando quali valori sono consentiti e definendo anche un messaggio di errore opportuno.

Ora per rendere più evidente le proporzioni tra soluto e solvente costruiamo dei grafici a torta. Osserviamo che nel foglio le due voci Soluto e Solvente non sono vicine e quindi prima di avviare la composizione automatica del grafico bisogna selezionare, tenendo premuto il tasto Ctrl, le celle Soluto e le celle Solvente con i relativi valori quindi si costruiranno i grafici come di solito. Durante la costruzione del grafico sono state selezionate le voci:

Serie dati in riga; Prima riga come didascalia; è stato chiesto di spostare la legenda in basso; dopo la costruzione del grafico è stato fatto un clic destro su di esso e dal menu contestuale è stato scelto Inserisci etichette dati; il carattere delle etichette è stato modificato dal menu contestuale ottenuto con un clic destro su di esse; poiché si è voluto dare un effetto di trasparenza all'area del grafico, è stato eseguito un clic destro sul grafico chiedendo di modificarlo e dopo un ulteriore clic destro verso l'area più esterna si è selezionato Formato area grafico → Trasparenza → Aumentare il valore della trasparenza (85%). Ecco l'effetto finale:

	A	B	C	D	E	F	G	H	I	J
2		SOLUZIONI								
4	INSERIRE I DATI NELLE CELLE EVIDENZIATE									
7	SOLUTO	SOLUZIONE	SOLVENTE	CONCENTRAZIONE		SOLUTO	SOLUZIONE	SOLVENTE	CONCENTRAZIONE	
8	20	400	380	5,00%		5	50	45	10,00%	
20	SOLUTO	SOLUZIONE	SOLVENTE	CONCENTRAZIONE						
21	90	300	210	30,00%						

20
380
SOLUTO SOLVENTE

5
45
SOLUTO SOLVENTE

90
210
SOLUTO SOLVENTE

Si intende salvare il foglio in formato html per essere pubblicato eventualmente su Internet.
Si sceglierà File → Salva con nome → si sceglie il percorso di destinazione del file → Si attribuisce il nome del file → dalla voce "Salva come" è possibile scegliere il formato che ci interessa tra tanti disponibili.

Esercizio n.10

Moti rettilinei

Vogliamo costruire un foglio per lo studio dei moti rettilinei uniformi ed uniformemente accelerati. Partiamo dalle leggi del moto uniforme: $s = s_0 + v_0*t$ e del moto uniformemente accelerato: $s = s_0 + v_0*t + ½*a*t^2$; vale inoltre la legge: v = v0 + a*t quando l'accelerazione a è costante. Notiamo che la legge del moto uniformemente accelerato si riduce a quella del moto uniforme quando a=0 quindi nel foglio useremo solo la formula più generale.

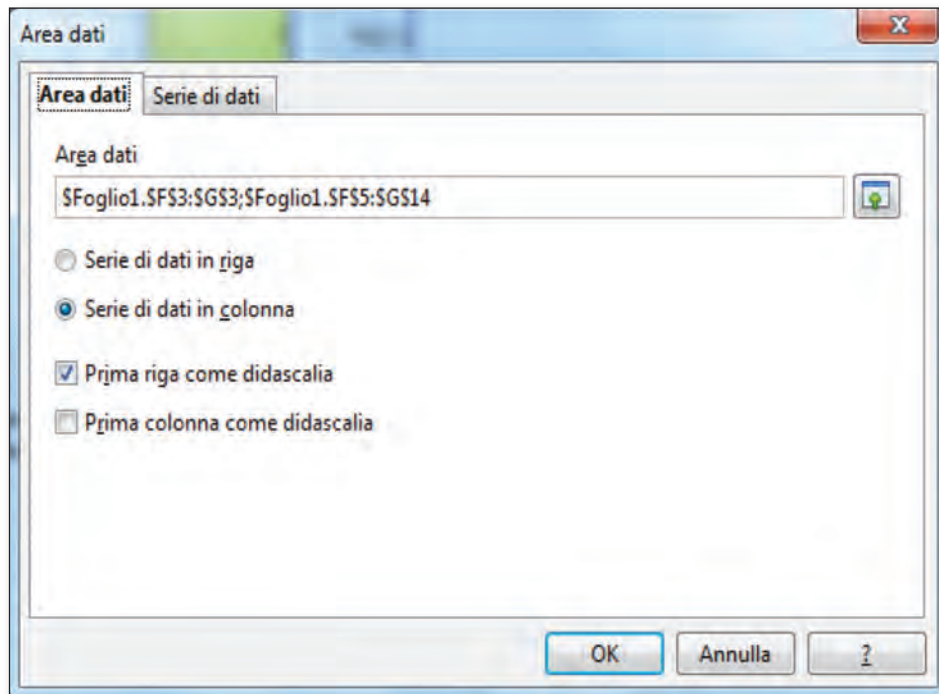
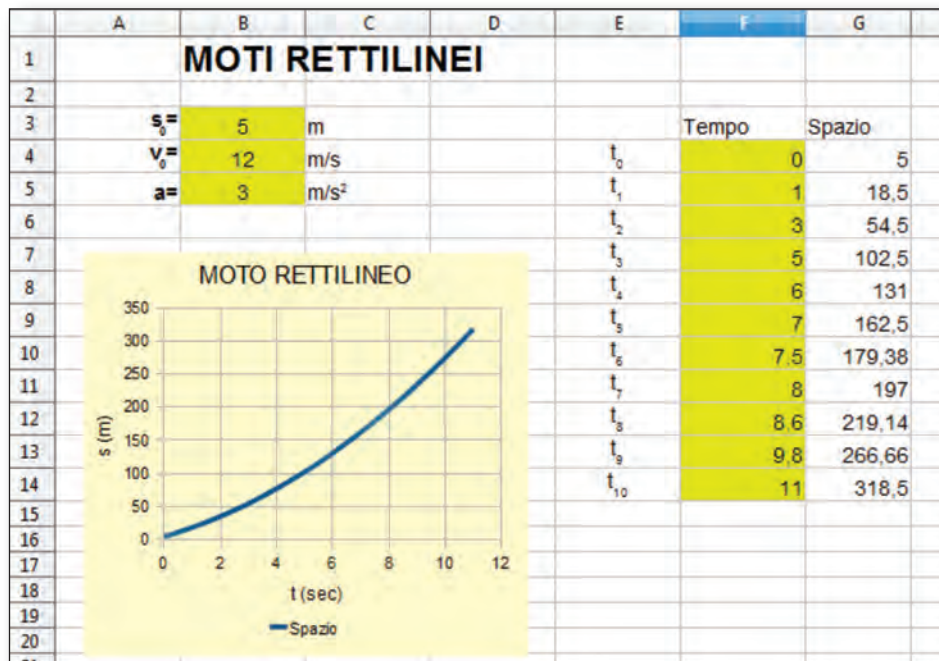
Si richiede di inserire i valori iniziali s_0, v_0, a e alcuni valori del tempo t (t>=0).

Il foglio calcola i corrispondenti valori di s e li rappresenta in un grafico. È interessante far valutare per quali valori delle costanti inserite, il grafico s è crescente o decrescente al variare di t, è una semiretta, è un arco di parabola, passa per l'origine degli assi.

La formula =B3+B4*F4+(B5*F4^2)/2 inserita in G4 non è in grado di genere valori attendibili quando viene ricopiata nelle celle sottostanti perché in tal caso si modificano anche i riferimenti alle celle B3, B4 e B5; prima di ricopiarla con il riempimento automatico è necessario bloccare i riferimenti alle suddette celle premendo contemporaneamente Maiusc+F4 sopra ai singoli riferimenti.

	A	B	C	D	E	F	G
1		**MOTI RETTILINEI**					
2							
3	s_0=	5	m			Tempo	Spazio
4	v_0=	12	m/s		t_1	1	=$B$3+$B$4*F4+($B$5*F4^2)/2
5	a=	3	m/s²		t_2	3	=$B$3+$B$4*F5+($B$5*F5^2)/2
6					t_3	5	=$B$3+$B$4*F6+($B$5*F6^2)/2
7					t_4	6	=$B$3+$B$4*F7+($B$5*F7^2)/2
8					t_5	7	=$B$3+$B$4*F8+($B$5*F8^2)/2
9					t_6	7,5	=$B$3+$B$4*F9+($B$5*F9^2)/2
10					t_7	8	=$B$3+$B$4*F10+($B$5*F10^2)/2
11					t_8	8,6	=$B$3+$B$4*F11+($B$5*F11^2)/2
12					t_9	9,8	=$B$3+$B$4*F12+($B$5*F12^2)/2
13					t_{10}	11	=$B$3+$B$4*F13+($B$5*F13^2)/2
14							

Ora rappresentiamo il grafico Spazio-Tempo di tipo xy personalizzandolo secondo i nostri gusti (clic destro sul grafico e modificare i diversi elementi).

Intanto è necessario aggiungere anche la situazione all'istante iniziale t=0: è sufficiente selezionare l'intervallo di celle da E4 a G13 (si indica con E4:G13) tenendo premuto il tasto sinistro del mouse mentre il cursore è all'interno dell'area selezionata è possibile trascinare nella direzione voluta. Ora però è necessario modificare l'estensione delle serie dei dati alle quali fa riferimento il grafico: clic destro sul grafico → Aree dati → Modificare la riga Area dati in modo che contenga solo il riferimento $Foglio1.$F$3:$G$14.

L'effetto finale è: Ora è possibile studiare il moto al variare di s_0, v_0 e dell'accelerazione a.

Esercizio n.11

Equazioni di 2° grado

Ora introduciamo un foglio per lo studio delle equazioni di 2° grado. Dopo l'inserimento delle etichette, scriviamo l'equazione di secondo grado in forma normale facendo attenzione a scrivere i coefficienti in celle separate rispetto ai caratteri alfanumerici.

Si faccia attenzione quando si inserisce il simbolo di uguaglianza; per evitare che il foglio elettronico lo interpreti come l'inizio di una formula è necessario premettere un apice cioè si scrive: "= .

Scriviamo le formule per calcolare il discriminante e le soluzioni.

	A	B	C	D	E	F	G
1		EQUAZIONI DI SECONDO GRADO					
2							
3	3	X²+	5	X +	-8	=	0

È necessario distinguere i diversi casi di **Δ >0, Δ=0 e Δ<0** poiché in ognuno di essi occorre che il foglio dia un diverso messaggio; ci serviamo della funzione **=SE.**

	A	B	C	D	E	F	G
1		EQUAZIONI DI SECONDO GRADO					
2							
3	4	X²+	1	X +	1	=	0
4							
5		**Δ =**	=C3^2-4*A3*E3				
6							
7		**X1 =**	=SE(C5<0;"Equazione impossibile";(-C3-RADQ(C5))/(2*A3))				
8							
9		**X2 =**	=SE(C5<0;"Equazione impossibile";(-C3+RADQ(C5))/(2*A3))				
10							

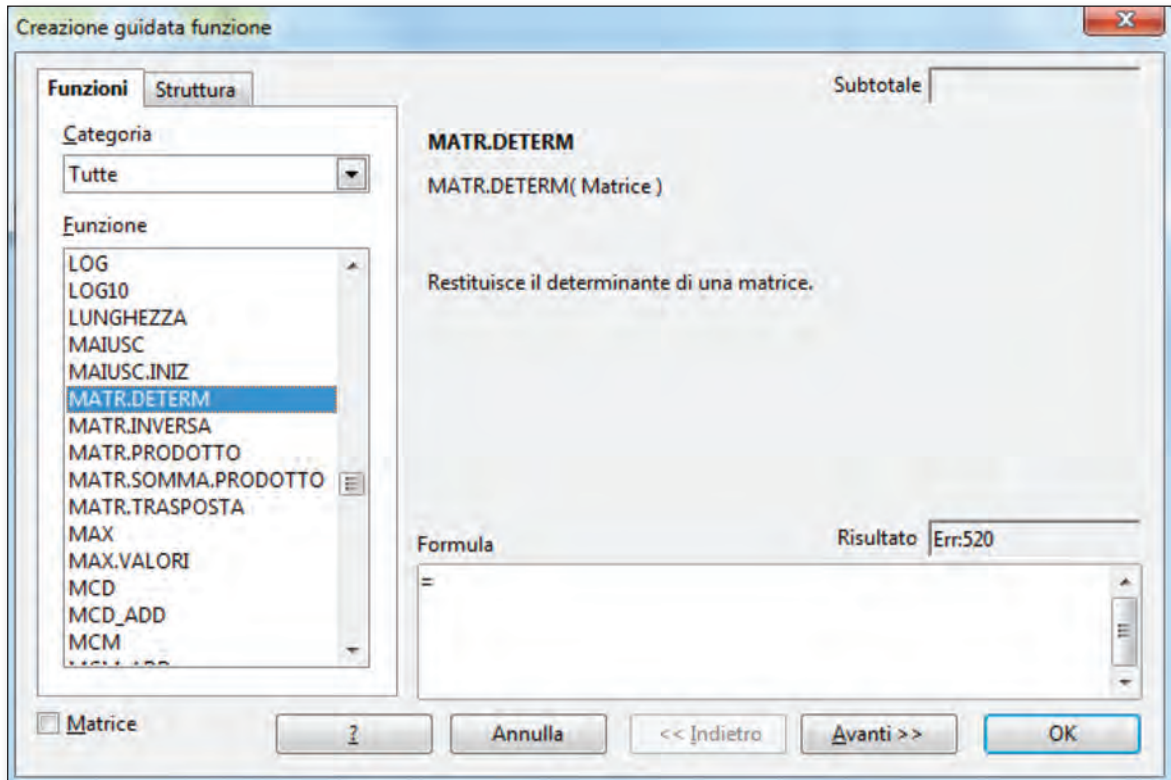
Sotto la condizione di Δ<0 il foglio deve scrivere "equazione impossibile" altrimenti deve calcolare le due radici x_1 e x_2. Vogliamo interpretare adesso l'equazione di secondo grado come l'intersezione tra una parabola e l'asse delle ascisse con un grafico Nell'intervallo I5:J8 è stato inserito un piccolo calcolo per determinare quanti valori di x bisogna considerare per passare dal valore minimo di x al valore massimo con un dato incremento.

Di questo si è tenuto conto nello stilare la tabella dei valori nell'intervallo A11:B22 scrivendo nella cella A12 la formula: "=J5" e nella cella A13 la formula: " =A12 + $J$7"; quest'ultima formula è stato ricopiata in basso per le celle necessarie. Nella cella B12 è stata scritta la formula =$A$3*A12^2+$C$3*A12+$E$3 che successivamente è stata ricopiata in basso. Blocchiamo le celle diverse da quelle di input (quelle evidenziate sono quelle di input) e salviamo in modalità compatibile con Excel XP dal percorso File → Salva con nome → Salva come.

Foglio elettronico ODF (.ods)
Modello di foglio elettronico ODF (.ots)
OpenOffice.org 1.0 - Foglio elettronico (.sxc)
OpenOffice.org 1.0 - Modello di foglio elettronico (.stc)
Data Interchange Format (.dif)
dBASE (.dbf)
Microsoft Excel 97/2000/XP (.xls)
Modello di documento Microsoft Excel 97/2000/XP (.xlt)
Microsoft Excel 95 (.xls)
Modello di documento Microsoft Excel 95 (.xlt)
Microsoft Excel 5.0 (.xls)
Modello di documento Microsoft Excel 5.0 (.xlt)
StarCalc 5.0 (.sdc)
Modello di documento StarCalc 5.0 (.vor)
StarCalc 4.0 (.sdc)
Modello di documento StarCalc 4.0 (.vor)
StarCalc 3.0 (.sdc)
Modello di documento StarCalc 3.0 (.vor)
SYLK (.slk)

Esercizio n.12

Sistemi Lineari

Il calcolo con le matrici è facilmente accessibile con il foglio elettronico e quindi si può applicare alla risoluzione dei sistemi lineari. L'esempio che riportiamo è per sistemi 3x3 ma è generalizzabile a tutti i sistemi lineari nxn. Si useranno le funzioni relative alle matrici accessibili dal tasto di creazione guidata funzione *fx* :

Le formule inserite sono le seguenti:

	A	B	C	D	E	F	G	H	I	J
1	SISTEMI LINEARI 3X3									
2										
3										
4		2	x+	1	y+	1	z=	5		
5										
6		1	x+	0	y+	-1	z=	3		
7										
8		1	x+	-1	y+	-2	z=	0		
9										
10										
11										
12		=$B$4	=$D$4	=$F$4						
13	Δ=	=$B$6	=$D$6	=$F$6	=	=MATR.DETERM(B12:D14)				
14		=$B$8	=$D$8	=$F$8						
15										
16								x=	Δx	=F18/F13
17		=$H$4	=$D$4	=$F$4					Δ	
18	Δx=	=$H$6	=$D$6	=$F$6	=	=MATR.DETERM(B17:D19)				
19		=$H$8	=$D$8	=$F$8						
20								y=	Δy	=F23/F13
21									Δ	
22		=$B$4	=$H$4	=$F$4						
23	Δy=	=$B$6	=$H$6	=$F$6	=	=MATR.DETERM(B22:D24)				
24		=$B$8	=$H$8	=$F$8						
25								z=	Δz	=F28/F13
26									Δ	
27		=$B$4	=$D$4	=$H$4						
28	Δz=	=$B$6	=$D$6	=$H$6	=	=MATR.DETERM(B27:D29)				
29		=$B$8	=$D$8	=$H$8						

Il risultato finale è:

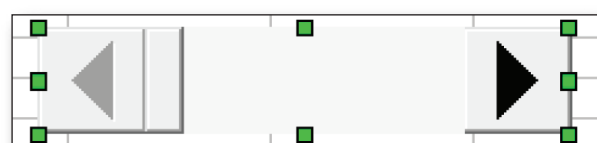
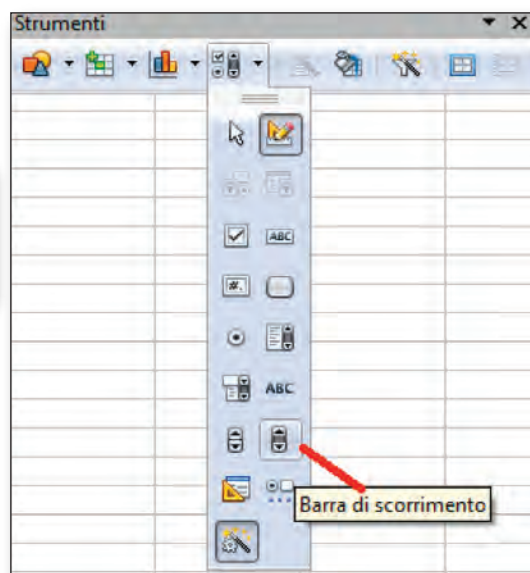
	A	B	C	D	E	F	G	H	I	J
1	SISTEMI LINEARI 3X3									
2										
3										
4		2	x+	1	y+	1	z=	5		
5										
6		1	x+	0	y+	-1	z=	3		
7										
8		1	x+	-1	y+	-2	z=	0		
9										
10										
11										
12		2	1	1						
13	Δ=	1	0	-1	=	-2				
14		1	-1	-2						
15										
16								x=	Δx	1
17		5	1	1					Δ	
18	Δx=	3	0	-1	=	-2				
19		0	-1	-2						
20								y=	Δy	5
21									Δ	
22		2	5	1						
23	Δy=	1	3	-1	=	-10				
24		1	0	-2						
25								z=	Δz	-2
26									Δ	
27		2	1	5						
28	Δz=	1	0	3	=	4				
29		1	-1	0						

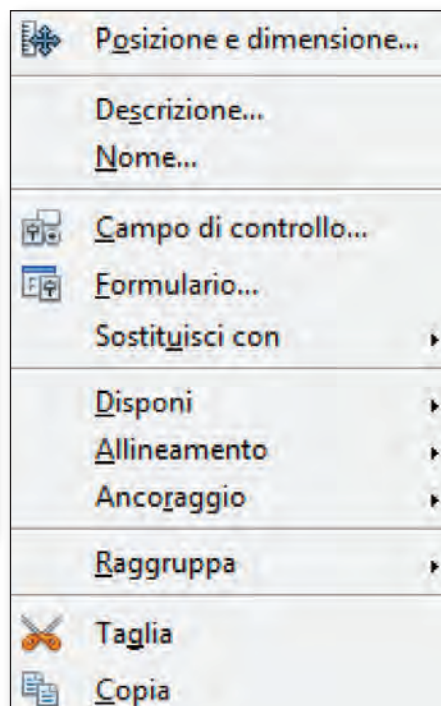
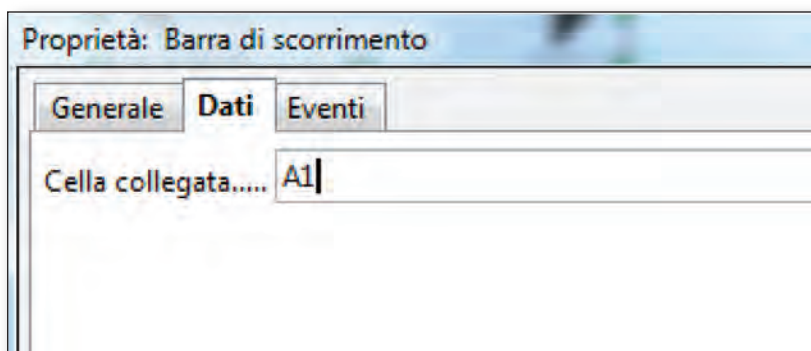
Esercizio n.13

La retta nel piano cartesiano

Ora presentiamo un foglio elettronico utile nello studio della retta nel piano cartesiano; attraverso di esso è possibile studiare come varia la retta al variare dei parametri m (coefficiente angolare) e q (intercetta). Useremo due barre di scorrimento per far assumere ad m e a q diversi valori. Per far questo è necessario visualizzare la barra degli strumenti "Strumenti" dal percorso: Visualizza → Barre degli strumenti → Strumenti

Bisogna agire sul pulsante "Campi di controllo" e selezionare il pulsate "Barra di scorrimento".

Ora basta trascinare il mouse in una regione qualsiasi del foglio di lavoro e viene disegnata una barra di scorrimento che deve però essere ancora configurata.

Iniziamo con un clic destro sulla barra di scorrimento per accedere alla personalizzazione dell'oggetto.

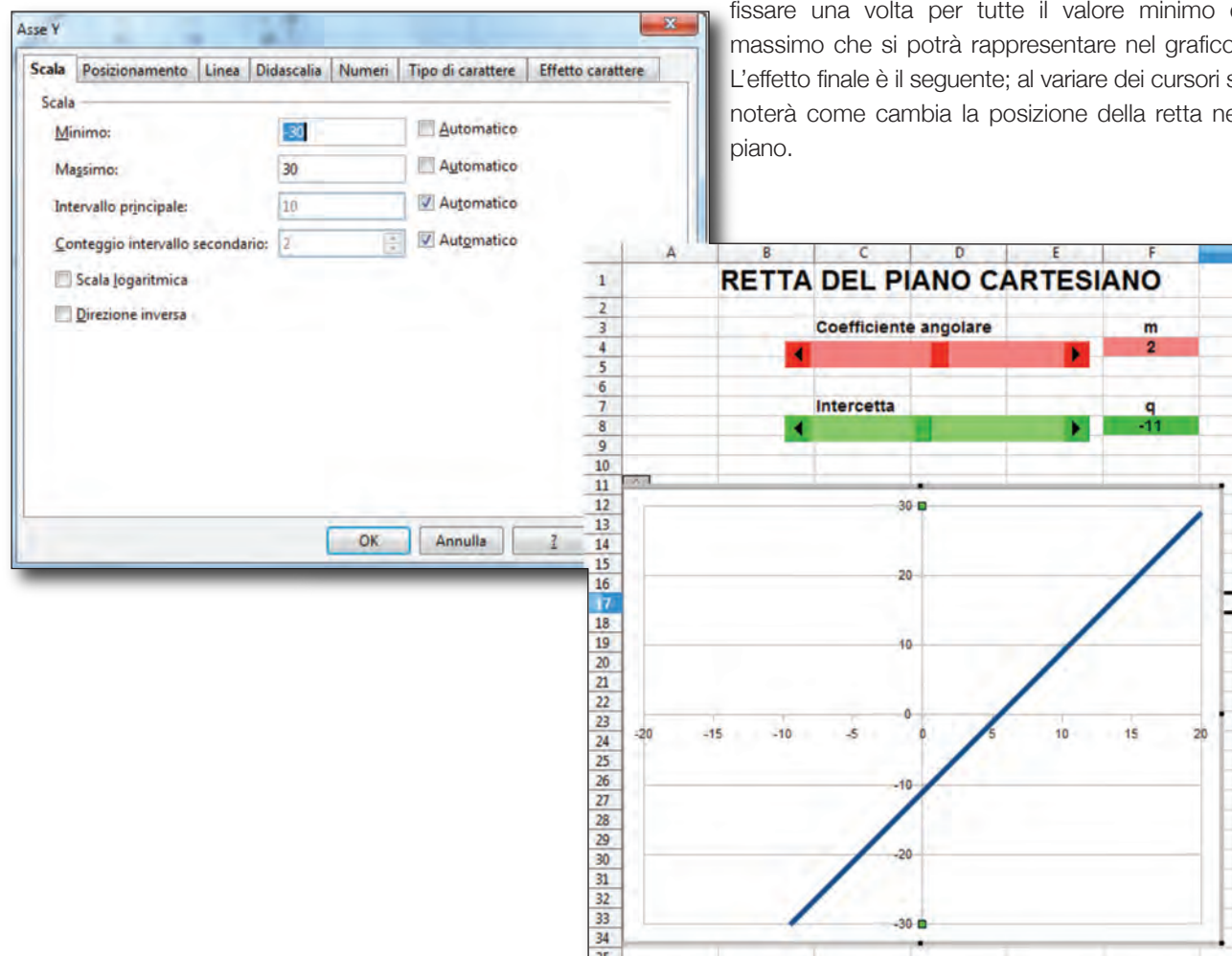
Dal menu contestuale scegliamo la voce “Campo di controllo”, impostiamo il valore minimo e massimo che si possono selezionare con la barra poi nella scheda “Dati” inseriamo l’indirizzo dove compariranno i valori che vengono selezionati con la barra.

Ora torniamo alla Barra degli Strumenti e clicchiamo nuovamente sul pulsante “Campi di controllo” quindi facciamo un clic sul pulsante . D’ora in avanti se variamo il cursore, nella cella A1 compariranno i diversi valori. Se intendiamo modificare le impostazioni della barra di scorrimento ritorniamo a cliccare sullo stesso pulsante .

Questi strumenti li usiamo adesso per costruire il nostro foglio per lo studio della retta.

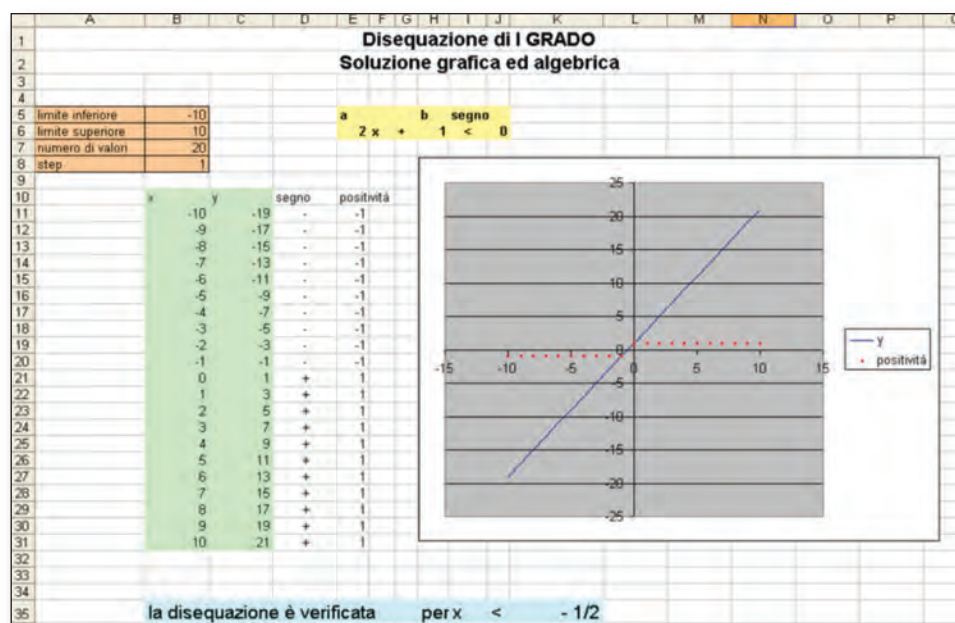
	A	B	C	D	E	F
1		RETTA DEL PIANO CARTESIANO				
2						
3			Coefficiente angolare			m
4						-15
5						
6						
7			Intercetta			q
8						44
9						
10						
11						

Queste barre di scorrimento sono inserite nel primo foglio della cartella che rinominiamo “Grafico” (clic destro sulla corrispondente linguetta del nome in basso a sinistra). Nel foglio successivo, che rinominiamo “Dati xy”, inseriamo i valori delle ascisse e delle ordinate dei punti appartenenti alla retta. Notiamo che le formule fanno riferimento ai valori di m e di q che si trovano nel primo foglio “Grafico” e che questi riferimenti sono stati impostati come assoluti (maiusc + F4).

	A	B
1	Dati retta	
2		
3	x	y
4	-100	=$Grafico.$F$4*A4+$Grafico.$F$8
5	-99	=$Grafico.$F$4*A5+$Grafico.$F$8
6	-98	=$Grafico.$F$4*A6+$Grafico.$F$8
7	-97	=$Grafico.$F$4*A7+$Grafico.$F$8
8	-96	=$Grafico.$F$4*A8+$Grafico.$F$8
9	-95	=$Grafico.$F$4*A9+$Grafico.$F$8
10	-94	=$Grafico.$F$4*A10+$Grafico.$F$8
11	-93	=$Grafico.$F$4*A11+$Grafico.$F$8
12	-92	=$Grafico.$F$4*A12+$Grafico.$F$8

Si è scelto di far variare il valore di x tra -100 e 100.
Ora selezioniamo i valori di x ed y e costruiamo il grafico xy.
Una volta costruito il grafico nel foglio "Dati" è possibile copiarlo e incollarlo nel foglio "Grafico".
Un clic destro in corrispondenza degli assi cartesiani ci permette di evitare che il foglio adatti automaticamente la scala e di fissare una volta per tutte il valore minimo e massimo che si potrà rappresentare nel grafico. L'effetto finale è il seguente; al variare dei cursori si noterà come cambia la posizione della retta nel piano.

Esercizio n.14

Soluzione grafica ed algebrica di una disequazione di I GRADO

Risultato finale

Aspetto matematico:
Una disequazione si dice di primo grado quando la variabile indipendente x compare con massima potenza uguale a 1

es: 2x + 3 > 0

Per risolvere una disequazione di I grado valgono le stesse regole delle equazioni di primo grado; solo che le equazioni di primo grado hanno come soluzione un solo valore (e cioè quel valore che sostituito alla x rende l'equazione vera), invece le disequazioni hanno come soluzione un insieme di valori.

Es: 2x + 3 > 0 è vera per x > -3/2

Quindi la soluzione e' l'insieme delle x maggiori di -3/2.
La soluzione può essere espressa nei seguenti modi:
{x ∈R / x > - 3/2}
oppure]-3/2, ∞[

Oppure

Quest'ultimo metodo di solito è il più usato (quando anche il valore terminale è compreso vi si mette un tondino)

Aspetto Informatico:
Prima di procedere allo svolgimento di questa esercitazione è necessario conoscere una funzione molto importante:
la funzione =SE() è una delle più importanti funzioni dei fogli elettronici in genere; infatti permette di far visualizzare in una cella due differenti valori a seconda che la condizione risulti vera o falsa.

La sintassi della funzione SE è la seguente:

=SE(condizione; se condizione vera; se condizione falsa)

Condizione: E' una espressione che come risultato deve restituire VERO o FALSO. In questo primo argomento vengono utilizzati gli operatori di uguaglianza in genere come:

"> ; < ; >= ; <= ; <> (diverso); ="

Es: A1>5 oppure B3=C5 oppure c1=D6*4 oppure F2="ok"

Da notare: il contenuto delle celle di tipo alfanumerico (Etichette) vengono indicate con "".

Se condizione vera / se condizione falsa: può contenere: un valore, una formula che viene inserito nella cella se il risultato della condizione è rispettivamente vero o falso.

Es: 5 oppure B2+C1 oppure C3/2 oppure "valore positivo"

Per provare …

Bisogna controllare la media dei voti di uno studente. Se la media (che è stata calcolata nella cella A5) è uguale o maggiore di 6 il ragazzo è PROMOSSO altrimenti è BOCCIATO

=SE(A5>=6;"PROMOSSO";"BOCCIATO")

Naturalmente questo è un esempio di uso abbastanza semplice della funzione SE ma nelle prossime esercitazioni vedremo come è possibile utilizzare questa funzione per controlli più complessi strutturando la funzione stessa con molti SE nidificati e con gli operatori booleani.

Procedimento:
Titolo dell'esercitazione

Rif. Cella	Cosa scrivo	Cosa visualizzo
A1	Disequazione di I GRADO	Disequazione di I GRADO
A2	Soluzione grafica ed algebrica	Soluzione grafica ed algebrica

Impostazione dell'intervallo per lo studio della espressione data

Rif. Cella	Cosa scrivo	Cosa visualizzo
A5	Limite inferiore	Limite inferiore
A6	Limite superiore	Limite superiore
A7	Numero dei valori	Numero dei valori
A8	Step	Step
B5	-10	-10
B6	10	10
B7	20	20
B8	=(B6-B5)/B7	1

Selezionare da A5 a B8 , attivare "bordi" → Tutti i bordi e colorare le celle a piacere.

Impostazione maschera di acquisizione Disequazione di I grado:

Rif. Cella	Cosa scrivo	Cosa visualizzo
E5	a	A
H5	b	B
I5	Segno	Segno
F6	x	X
G6	‘+	+
I6	<	<
J6	0	0

Selezionare da E5 a J6 e colorare le celle con un colore a scelta. Ridimensionare le celle dalla colonna E alla colonna J per ottenere il risultato della Fig.1.

Impostazione della tabella

Rif. Cella	Cosa scrivo	Cosa visualizzo
B10	X	x
C10	Y	y
D10	Segno	Segno
E10	Positività	Positività
B11	=B5	-10
B12	=B11+b8	-9
B13-B31	Trascinare	Da -8 a 10
C11	=$E$6*B11+$H$6	-19
C12-C31	Trascinare	Da -17 a 21
D11	=SE(C11<0;"-";"+")	Il segno "+" oppure "-"
E11	=SE(D11="+";1;-1)	Il valore 1 oppure -1
D12-D31	Trascinare	
E12-E31	Trascinare	

Selezionare da B10 a E31 , attivare "bordi" → "Tutti i bordi" e colorare le celle con un colore a piacere.

Soluzione grafica
Selezionare 3 colonne della tabella:
Selezionare da *B10* a *C31* e premuto il tasto "**CTRL**" selezionare da *E10* a *E31*.
Attivare la creazione guidata grafico e scegliere tra i vari **Tipi di grafico** quello a **Dispersione.**

Soluzione algebrica

Rif. Cella	Cosa scrivo	Cosa visualizzo
B35	La disequazione è verificata per x	La disequazione è verificata per x
J35	=J6	<
k35	=-H6/E6	-0.5

Selezionare la cella K35 e modificare il tipo di numero come "frazione".
Selezionare da B35 a K35 e cambiare il colore a tuo piacere.

Avendo impostato il foglio in questo modo ora sarai in grado di risolvere qualsiasi disequazione di I Grado. Dovrai, infatti, semplicemente modificare i coefficienti a,b ed il segno e il risultato sarà immediato.

9. Strumenti di presentazione (OpenOffice Impress)

Autore: **Salvatore Madaro**

Competenze	Abilità	Conoscenze
Utilizzare, con autonomia operativa ed organizzativa, strumenti di comunicazione visiva e multimediale, anche con riferimento alle strategie espressive e agli strumenti tecnici della comunicazione in rete.	*Utilizzare i principali software per la produttività individuale.*	*Software di utilità e software applicativi.*
	Raccogliere, organizzare e rappresentare informazioni.	*Metodi di rappresentazione di informazione di documentazione.*
	Rappresentare ed elaborare i risultati delle misure di grandezze fisiche utilizzando strumenti informatici.	*Strumenti di presentazione*

Progettare una presentazione

Un ottimo programma di presentazioni multimediali totalmente gratuito è **Impress** del pacchetto OpenOffice o di LibreOffice. In questi appunti ci riferiamo alla versione 3.3.0 di OpenOffice scaricabile dal sito http://www.openoffice.org/it/download/3.3.0/download330.html.

Impress consente di costruire delle buone presentazioni di tesine, di argomenti di diverso genere, di prodotti, ecc. con un minimo sforzo e con risultati davvero interessanti.

Quando costruiamo le diapositive (slide) di una presentazione è bene tener presente che è necessario non eccedere negli elementi grafici , nel numero dei colori oppure nelle animazioni e transizioni altrimenti, invece di catturare l'attenzione si ha esattamente l'effetto opposto.

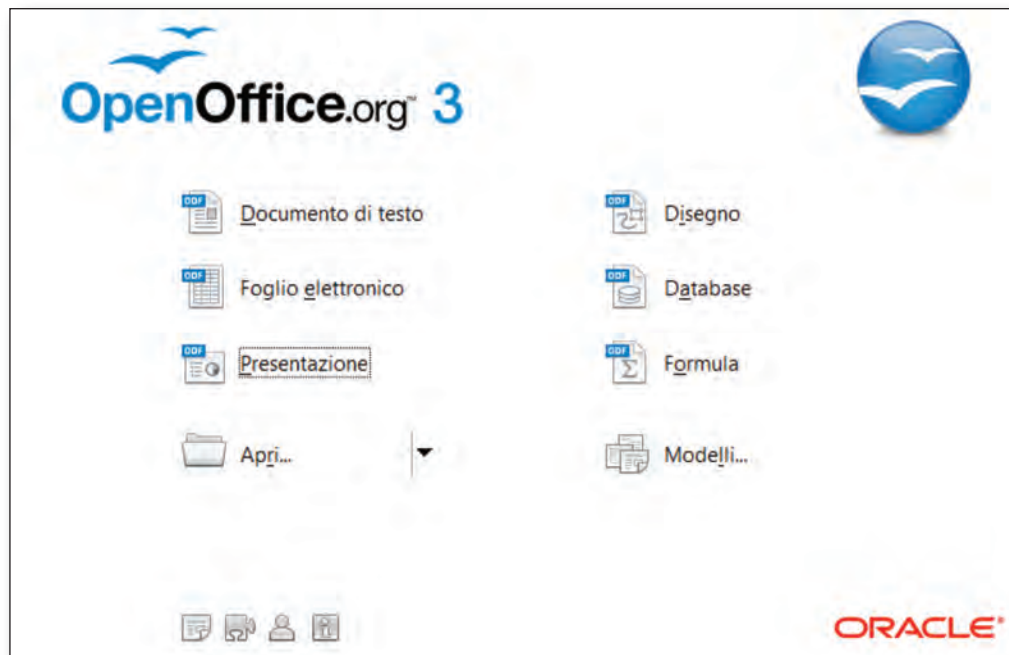
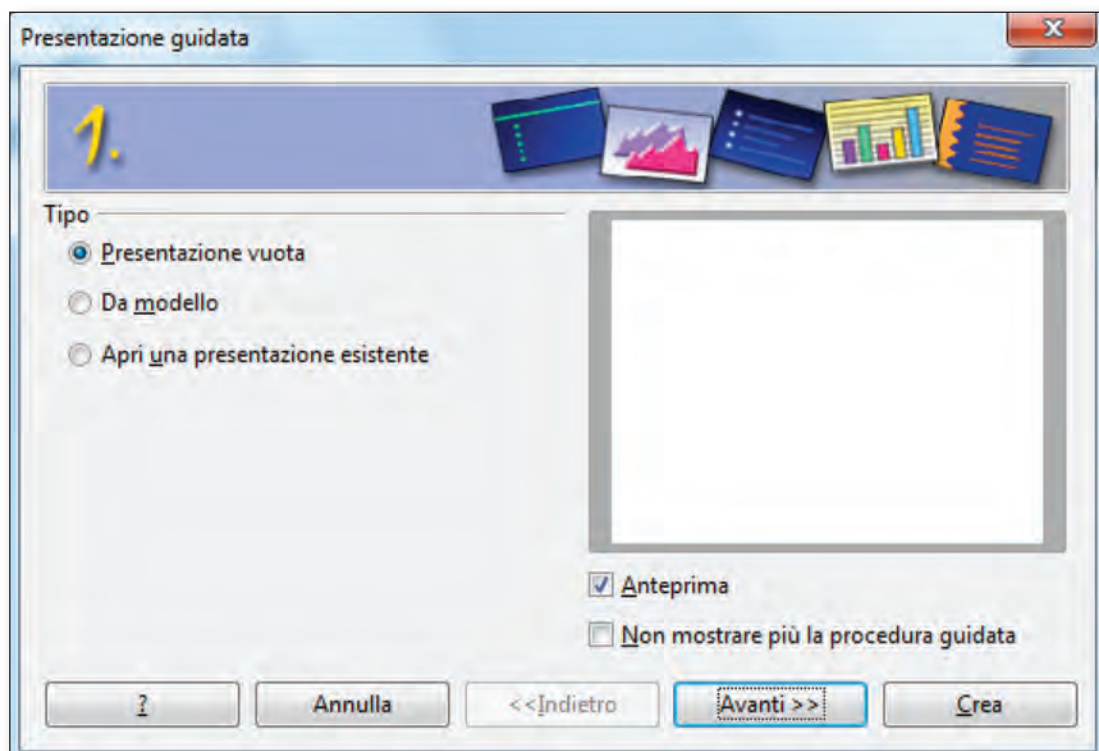
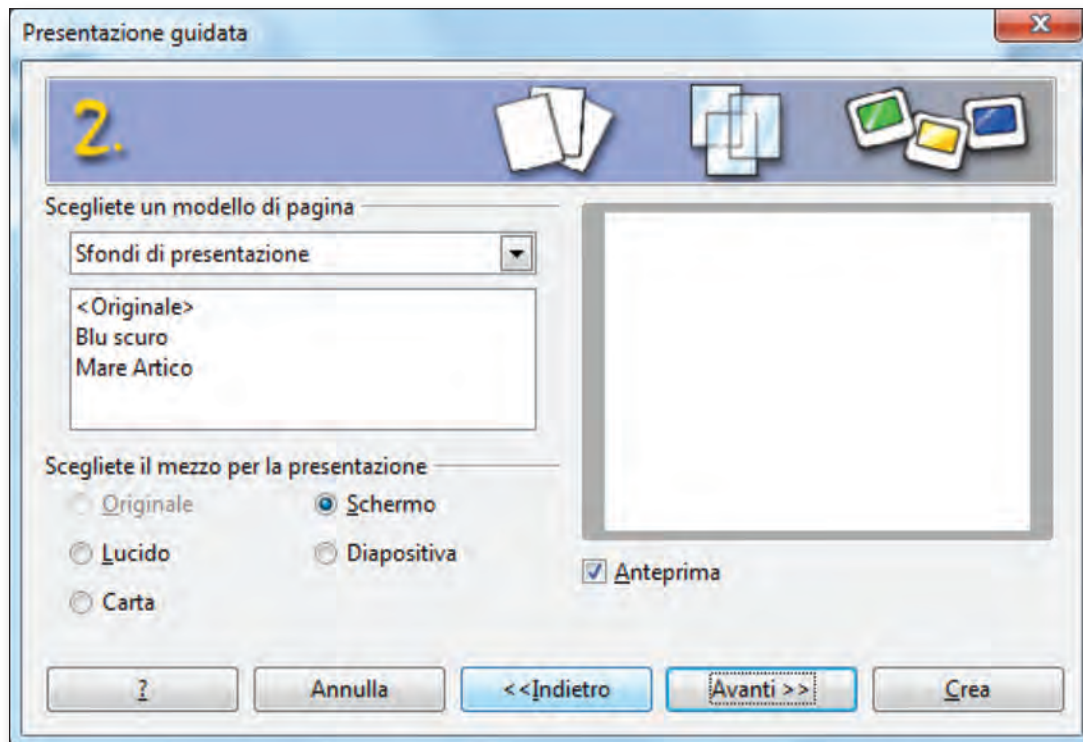
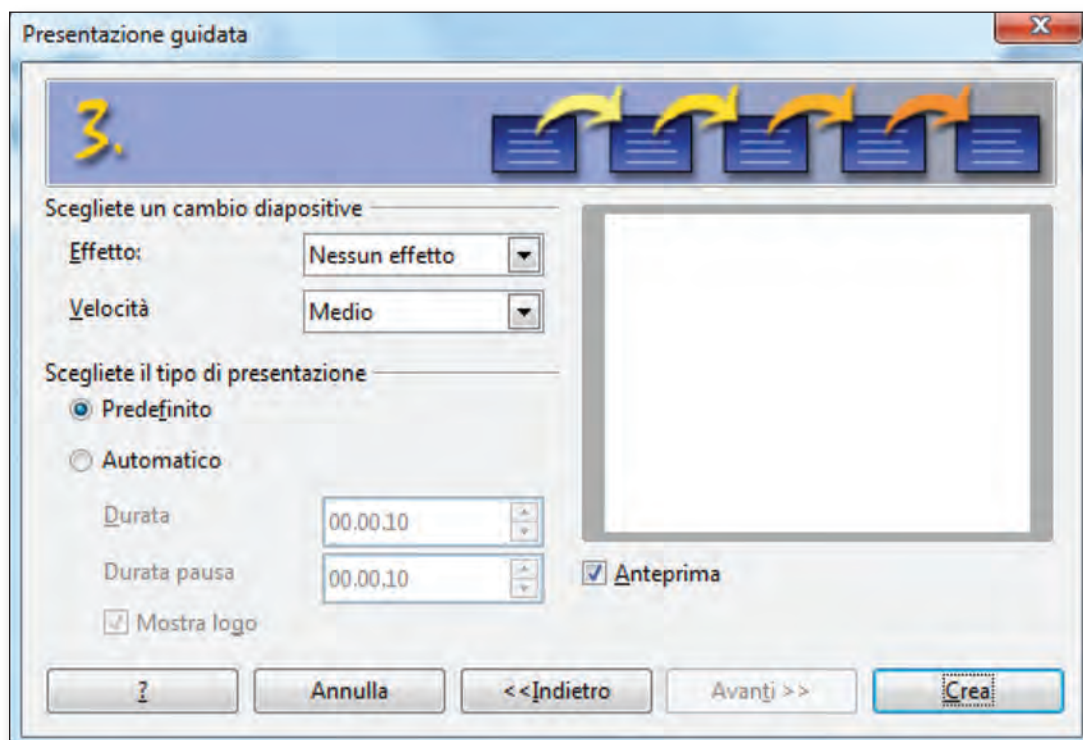
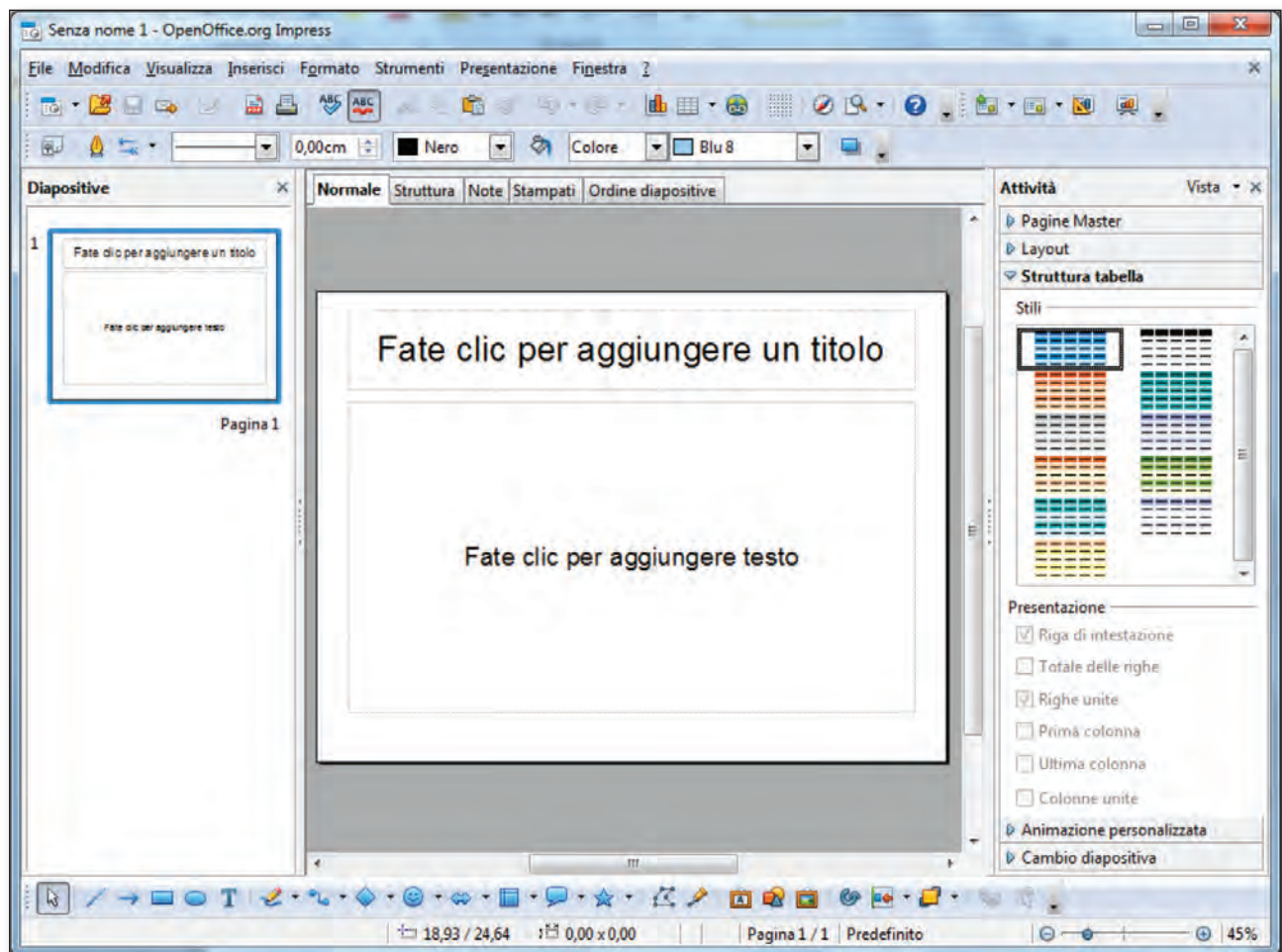
Un'altra cosa da considerare è che una diapositiva non deve essere una pagina carica di troppo testo; è necessario che contenga una sintesi estrema dei concetti e alcuni elementi grafici attinenti a quanto è scritto. Il relatore non deve leggere dalle diapositive come fossero dei fogli di testo ma deve usare le diapositive per riassumere agli ascoltatori i concetti essenziali del discorso.

È utile tenere sempre presente a quale pubblico è destinata la presentazione per decidere sulla scelta degli sfondi, dei caratteri, dei colori, degli effetti ….

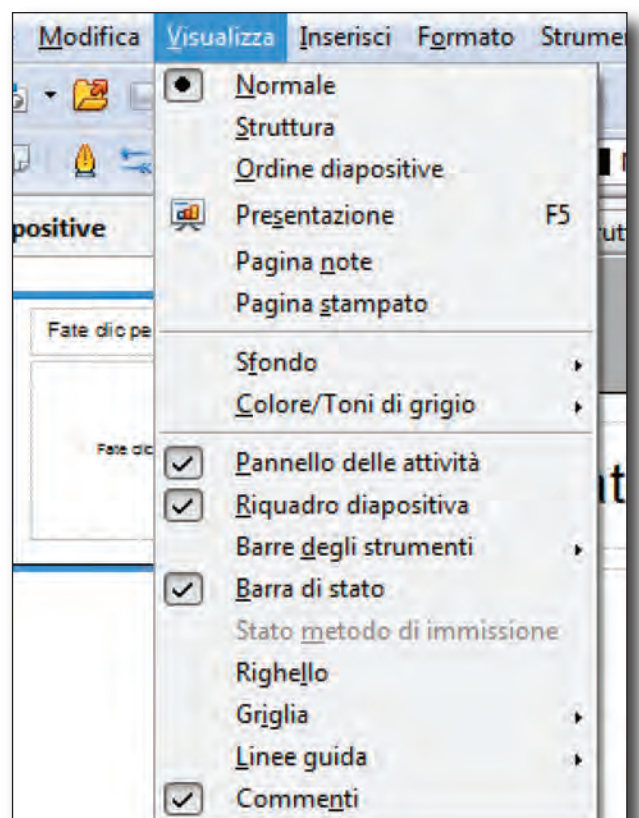
I colori sono una componente essenziale della presentazione poiché evocano delle sensazioni, dei sentimenti che potrebbero essere in conflitto con quello che vogliamo comunicare. Ad esempio il **nero** evoca noia, dolore, morte; il **marrone** ricorda la terra, la semplicità, i luoghi aperti; il **blu** suscita sentimenti di pace, tranquillità, sicurezza, confidenza; il **viola** stimola la spiritualità, il senso del mistero, della ricchezza; il **verde** ricorda la natura, l'ambiente, la salute ma anche i rettili e gli insetti; il **grigio** dà il senso della conservazione, dell'affidabilità, della sicurezza e dell'equilibrio; il **rosso** ricorda la passione, l'eccitazione, l'amore, il calore e l'aggressione; **l'arancione** stimola il senso del calore, della ricchezza; il **giallo** infonde ottimismo, felicità e stimola l'immaginazione; il **bianco** suggerisce la purezza, il rispetto e la semplicità.

Da evitare gli accostamenti **rosso–verde**, **arancione–blu** e **rosso-blu** così come sono da evitare i caratteri con molte “grazie” cioè troppo elaborati.

Passiamo all'opera: Impostazioni Iniziali

Per costruire una presentazione, all'avvio di OpenOffice è necessario scegliere il programma **Impress** (Presentazione nella versione italiana),

Sarà possibile quindi scegliere se aprire una precedente presentazione, usare un modello predefinito per inserire i nostri dati e creare così una presentazione sotto la guida dello stesso software oppure si può optare per la creazione di una presentazione vuota dove il nostro estro potrà dare sfogo a tutta la propria energia! Noi scegliamo quest'ultima opzione.

Ancora qualche clic nella schermata successiva per adattare la nostra presentazione al tipo di supporto o per decidere se usare un modello di pagina preesistente

Nella schermata successiva, se lo vogliamo, possiamo impostare gli effetti di transizione tra una diapositiva e la successiva ed i relativi tempi:

Se si è deciso di non usare modelli, pagine o effetti preimpostati lo schermo che appare è il seguente:

Si riconoscono, dall'alto in basso, la barra del titolo, la barra dei menu, le barre degli strumenti, il riquadro diapositive (a sinistra) nella quale è visualizzata la sequenza delle diapositive, l'area di lavoro (al centro) nella quale è visualizzata la diapositiva attiva sulla quale si sta lavorando, il pannello delle attività (a destra) che contiene gli strumenti per modificare le diapositive ed applicare gli effetti di animazione e di transizione, la barra del disegno (in basso) che contiene gli strumenti per inserire degli elementi grafici all'interno della diapositiva corrente ed infine la barra di stato nella quale è riportata la posizione di un oggetto all'interno della diapositiva e le relative dimensioni, il numero della diapositiva corrente sul totale che compongono la presentazione e un comodo cursore per lo zoom.

Nell'area di lavoro è possibile cambiare "Vista" cliccando semplicemente sulle linguette poste sopra di essa: Normale, Struttura, Note, Stampati, Ordine diapositive.

Lo stesso effetto lo possiamo ottenere selezionando nella barra dei menù la voce "Visualizza". Tali viste consentono di lavorare con diversi livelli di dettaglio sulla presentazione.

Normale: è la visualizzazione più comune con la diapositiva corrente visualizzata al centro e la sequenza di tutte le diapositive visualizzata sul lato sinistro.

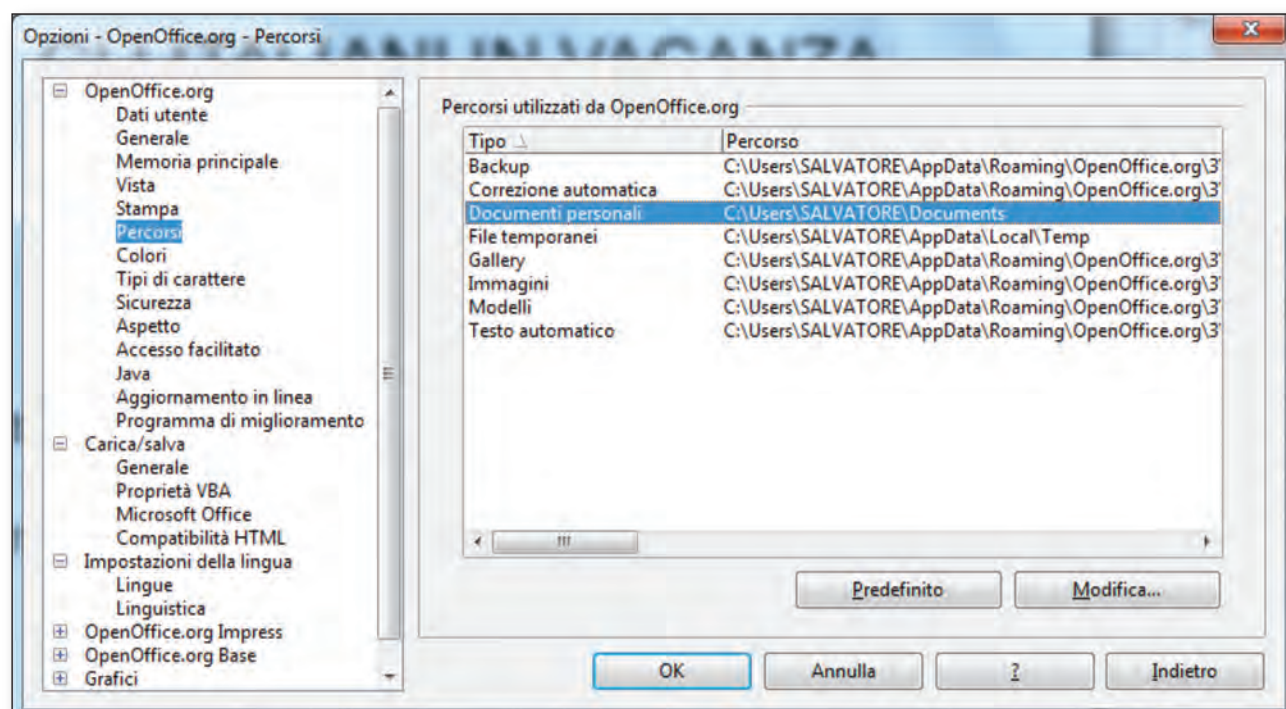
Struttura: mostra l'elenco delle diapositive (che appaiono come icone)della presentazione consentendo sia di cambiare l'ordine che di modificare i relativi titoli.

Note: è possibile aggiungere, modificare o visualizzare le note per il relatore.

Stampati: permette di vedere in anteprima come saranno stampate le diapositive e modificare la disposizione finale degli elementi che compongono una pagina (n° di diapositive per pagina, intestazione, piè di pagina ecc.)

Ordine diapositive: mostra tutte le anteprime delle diapositive consentendo applicare sfondi ed effetti di transizione simultaneamente a tutte o a un certo numero di esse selezionate o mediante ctrl + click, o con il tasto Maiuscolo + click oppure con il trascinamento del cursore su un gruppo di diapositive contigue.

Se non riteniamo indispensabile avere sullo schermo alcune barre degli strumenti o dei pannelli poiché riteniamo di non doverne fare uso, è possibile nasconderli mediante il corrispondente pulsante di chiusura x (se presente) oppure dal percorso: Barra dei menù → Visualizza e togliere il segno di spunta dalle voci che non interessano; si ottiene così più spazio per lavorare più comodamente con le diapositive.

I più esigenti potranno modificare altre impostazioni di Impress dal seguente percorso: Barra dei menù → Strumenti → Opzioni; si apre una finestra dalla quale si possono modificare diversi parametri che incidono sul funzionamento del programma.
Tra queste voci c'è anche "Percorsi" dalla quale è possibile cambiare la cartella dove Impress salva automaticamente i propri lavori.

Costruire Diapositive

Inizialmente manteniamoci nella vista normale e avviamo la costruzione della prima diapositiva: la diapositiva titolo.
Come si vede dall'immagine, sono richiesti un titolo per la prima diapositiva (che sarà anche quello della dato in automatico
alla presentazione) e del testo, detto comunemente sottotitolo, nel quale inseriremo l'autore, la data, il contesto per il quale tale presentazione è stata creata, ecc. Si avrà cura di scegliere un titolo e dei dati significativi e coerenti con quanto sarà proiettato.

Ora, a titolo di esempio, vogliamo creare una presentazione sulle abitudini degli italiani relativamente alle proprie vacanze.

GLI ITALIANI IN VACANZA

Breve sintesi delle abitudini degli italiani nella scelta delle proprie vacanze.
A cura di Salvatore Madaro per Book in Progress

Osserviamo che nella diapositiva sono presenti due oggetti; in questo caso sono due caselle di testo nelle quali sono già stati predisposti la dimensione ed il tipo di carattere con l'invito ad inserire del testo. È sufficiente cliccare all'interno e scrivere il titolo ed il sottotitolo.

Tutte le diapositive avranno un titolo ed è bene che sia diverso da slide a slide perché sarà più semplice riconoscerle se vogliamo agire in seguito sul loro ordine o vogliamo applicare degli sfondi o degli effetti.

Naturalmente una slide così piatta non è gradevole e non suscita alcuna sensazione, non evoca nulla!

Se aggiungiamo uno sfondo e qualche elemento grafico, sicuramente l'effetto sarà più gradevole; vediamo come fare.

Nella barra del disegno (in basso) ci sono degli strumenti che permettono di inserire delle forme predefinite , delle frecce, dei fumetti oppure dei disegni o delle foto presenti in un file.

Per inserire una forma basta sceglierla dal pulsante corrispondente, selezionarla con un click, passare nell'area di lavoro, tenere premuto il tasto sinistro del mouse e trascinare fino alla dimensione desiderata . È possibile comunque modificare successivamente la dimensione dell'oggetto selezionato agendo sulle relative maniglie di dimensionamento che compaiono ai lati e sui margini superiore ed inferiore dell'oggetto quando questo viene selezionato con un click del mouse.
Dopo aver selezionato del testo si può modificarlo agendo sugli strumenti della relativa barra dove si sceglie il tipo di carattere, le dimensioni, il grassetto, il corsivo, il sottolineato, l'ombreggiato, l'allineamento orizzontale, il colore, gli eventuali effetti ecc.

Nell'esempio seguente è stato assegnato al testo l'effetto ombreggiato ed il contorno.
Le foto sono state spostate dietro al testo selezionando l'immagine, click destro, menù contestuale, Disponi, Porta in fondo.

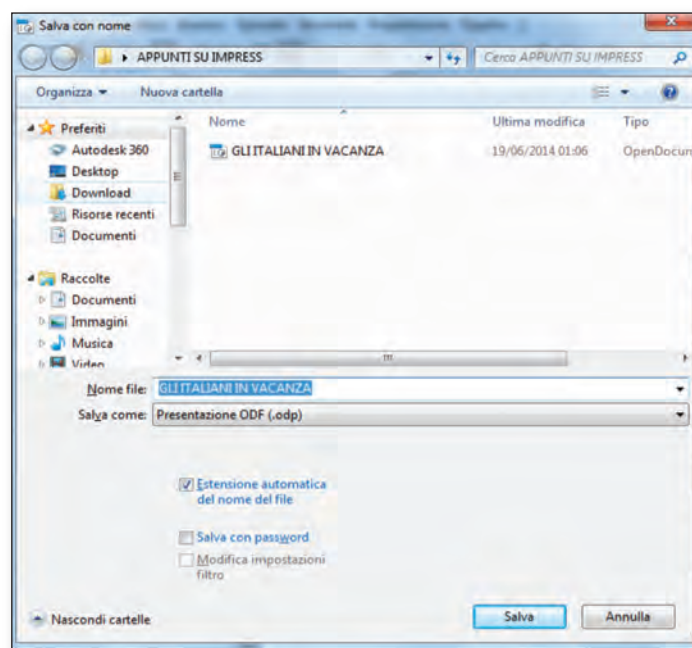
Anche se i gusti personali possono cambiare da soggetto a soggetto, osservate che la presenza di due foto, qualche effetto sul testo e la presenza qualche forma predefinita, migliora la comunicazione dei contenuti della diapositiva. State attenti però a non esagerare.

Salvare la Presentazione

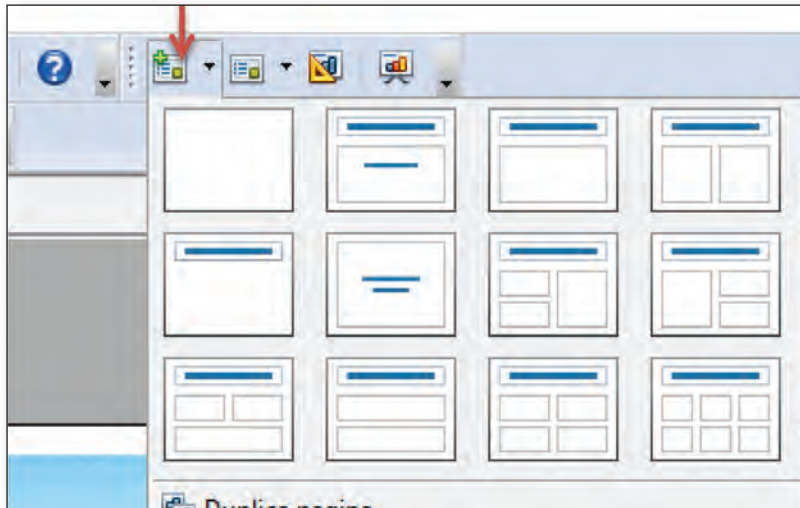
È importante salvare di tanto in tanto il nostro lavoro su disco o su pennetta USB per evitare che uno sbalzo di tensione o un blocco improvviso del PC ci faccia perdere tutto quello che abbiamo realizzato.

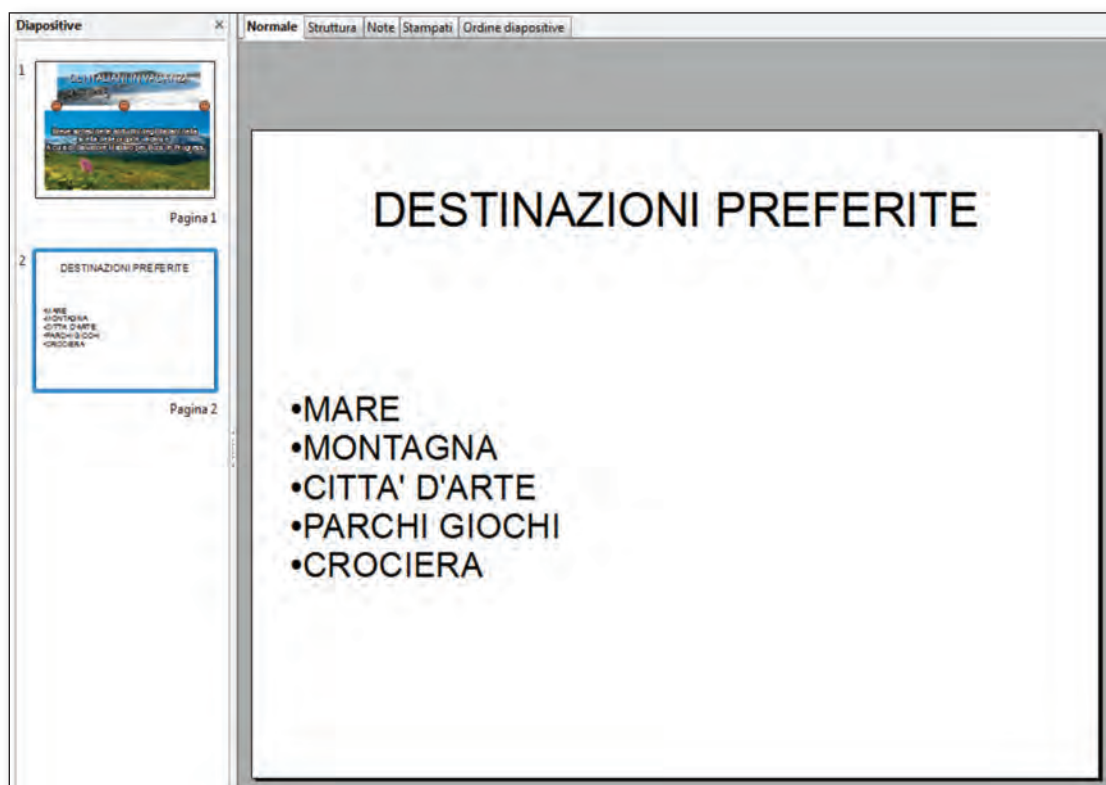
Il pulsante "Salva" permette di salvare la presentazione ; naturalmente anche se è rappresentato un floppy disk (che sembra appartenere all'era preistorica) il salvataggio viene eseguito su una qualsiasi memoria di massa disponibile (hard-disk, disco esterno, pennetta USB o altro).
La prima volta che si salva il file , viene chiesta la posizione dove scrivere; selezioniamo il percorso di destinazione, assegniamo il nome e scegliamo il formato del file.
Se in seguito vogliamo salvare il file con un altro nome o in un altro percorso è necessario cliccare sul pulsante "Salva con nome" Salva con nome... che si trova nel menù "File" altrimenti, selezionando il pulsante "Salva", programma sovrascriverà sul file precedente la nuova versione.

Osserviamo che tra i vari formati nei quali Impress può salvare, c'è quello proprietario (cioè .odp) e quello compatibile con il suo diretto concorrente (ossia . ppt).

C'è una parziale compatibilità tra i file di Impress e quelli di PowerPoint nel senso che un programma apre i file dell'altro però non è garantito che tutti gli elementi grafici, gli effetti, le animazioni vengano conservati al 100%. È possibile salvare anche nel formato PDF; in questo formato la presentazione perde tutti gli effetti di animazione e transizione che vedremo in seguito ma è riconoscibile senza problemi da qualsiasi READER PDF ed occupa minore spazio rispetto alla normale presentazione.

GLI ITALIANI IN VACANZA	27/06/2014 15:18	OpenDocument - ...	1.101 KB
GLI ITALIANI IN VACANZA	27/06/2014 14:42	Adobe Acrobat D...	595 KB

I layout delle diapositive

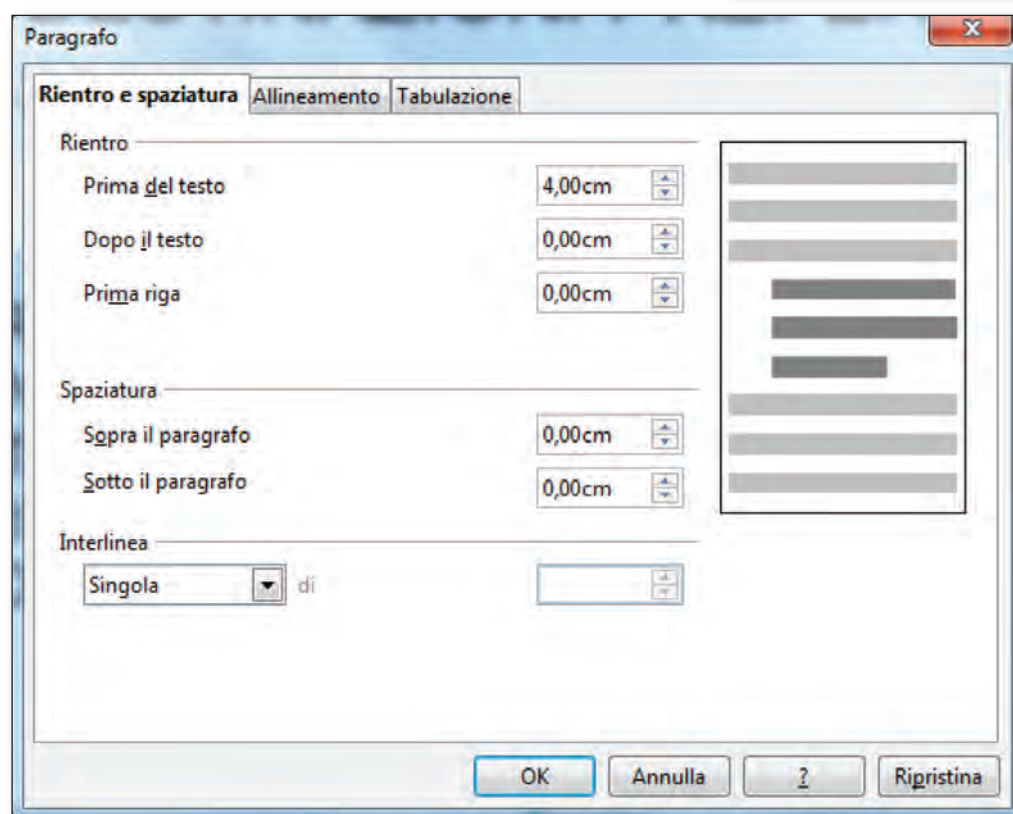
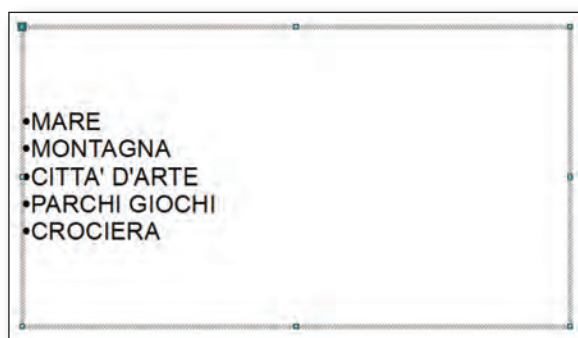
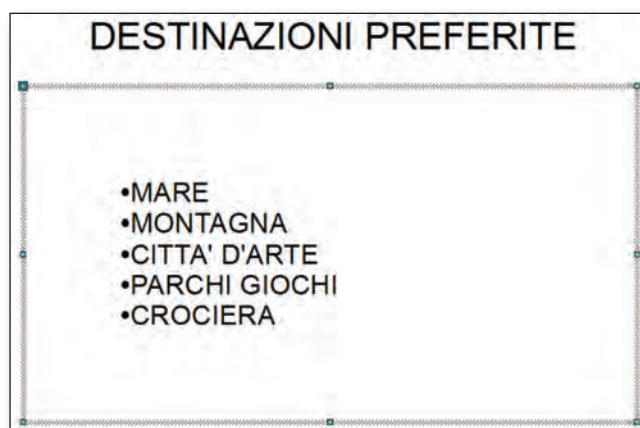
Adesso che la prima diapositiva titolo è stata realizzata, vogliamo continuare a creare la altre diapositive nei diversi formati disponibili. Innanzitutto dobbiamo aggiungere una nuova slide con l'apposito pulsante "Aggiungi Pagina" che consente anche di scegliere il suo layout premendo la freccetta al lato del pulsante che apre una scheda contestuale.

È possibile scegliere tra le alternative proposte oppure costruirne una da zero aggiungendo mano a mano le caselle di testo e gli elementi grafici necessari. Cominciamo ad inserire una slide con titolo e testo:

Inseriamo il titolo della diapositiva e scriviamo del testo nella parte inferiore; per migliorare la lettura delle singole voci applichiamo un elenco puntato:

L'elenco puntato si ottiene selezionando la casella del testo e cliccando sul pulsante Il testo inizialmente era allineato al centro; l'allineamento a sinistra si è ottenuto premendo il pulsante .

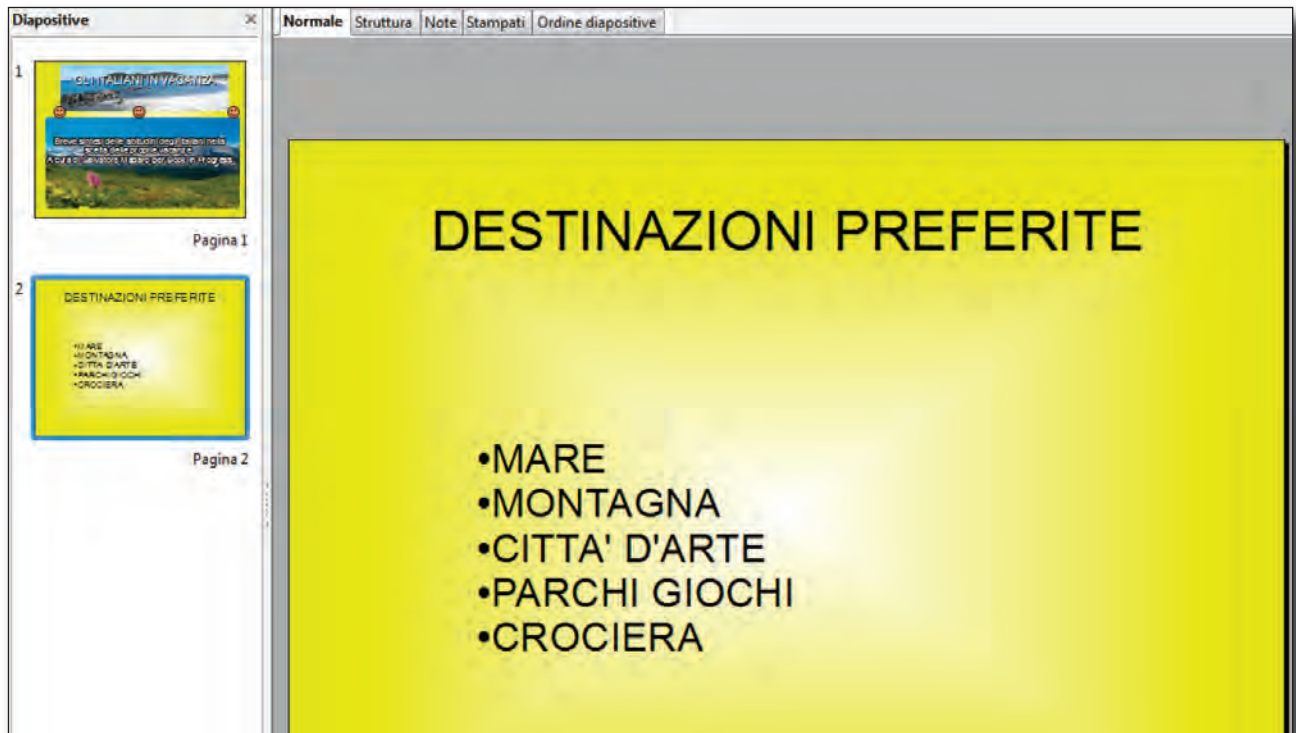
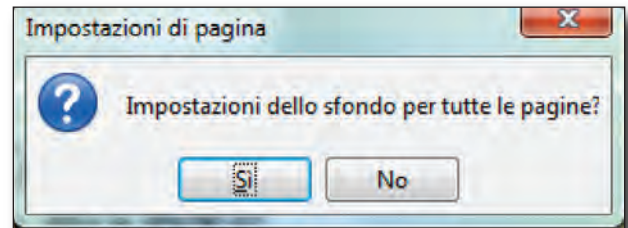
Da notare che l'elenco con l'anteprima delle slide presente a sinistra si è arricchito della diapositiva che abbiamo aggiunto.

Poiché il risultato non ci soddisfa, vogliamo allontanare il testo dal bordo sinistro e aggiungere uno sfondo a tutte le diapositive che costruiremo a partire da quella corrente.

Spostiamo il testo mediante il pulsante “Paragrafo” ; selezioniamo il testo cliccando sul bordo della casella di testo in modo che siano visualizzate le maniglie di dimensionamento.

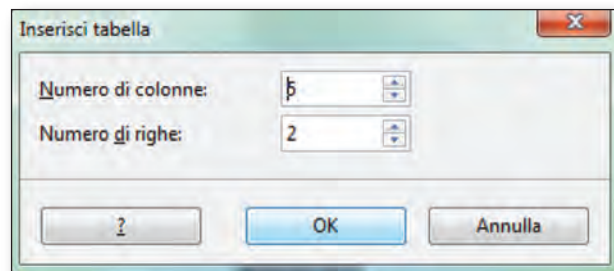
Clicchiamo sul pulsante “Paragrafo”, si apre la corrispondente scheda dalla quale si può impostare il rientro desiderato:

ed ecco il risultato:

Inserire uno sfondo alle diapositive

Il percorso è : Barra dei menù → Formato → Pagina →Sfondo e scegliere tra le alternative disponibili. è possibile impostare lo sfondo o solo per la diapositiva corrente e oppure per tutte le slide della presentazione.

Ed ecco il risultato:

Per rendere più gradevole la diapositiva aggiungiamo una immagine attinente all'argomento, tramite il pulsante "Da file" presente sulla barra Disegno.

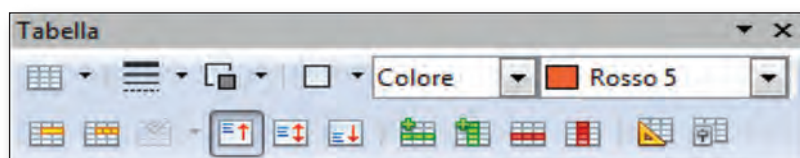
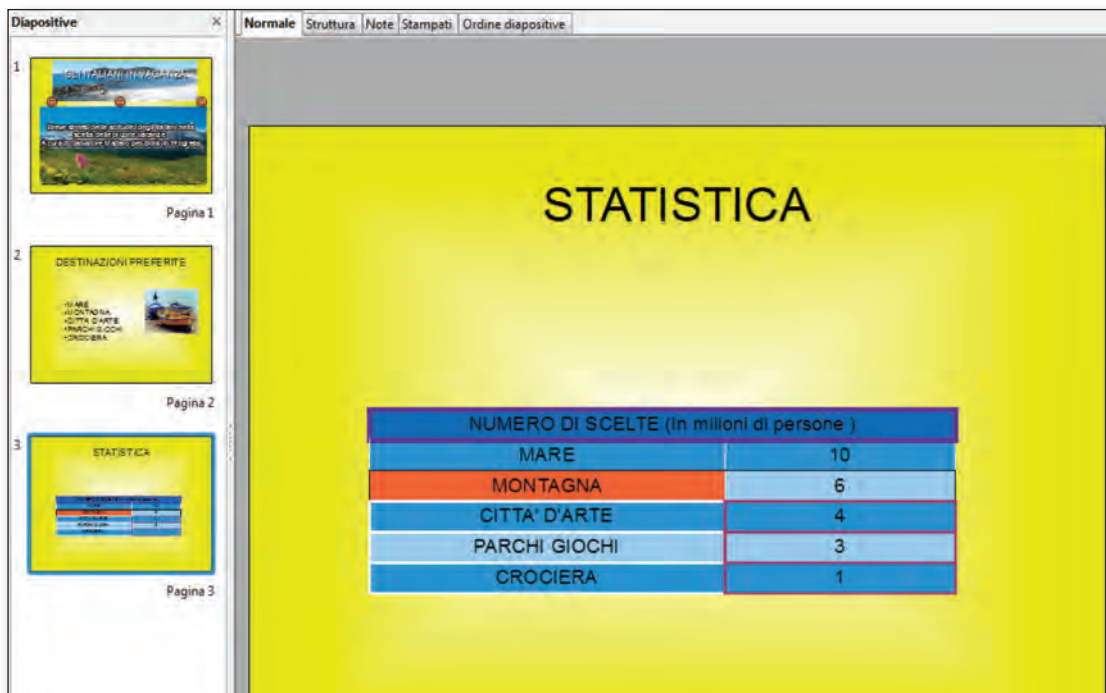
Adesso aggiungiamo una diapositiva nella quale inseriremo una tabella con i dati statistici relativi all'argomento (inventeremo dei dati fittizi). Tramite il pulsante "Pagina" scegliamo il layout "Titolo, Contenu- to" e ci ritroveremo nella seguente situazione:

Le tabelle

Naturalmente inseriremo il titolo della diapositiva e, dal quadrato al centro dell' area "Contenuto", selezioniamo l'icona corrispondente alla **tabella** (il quadrante in alto a sinistra); si apre una scheda dove è necessario inserire il numero di righe e di colonne che devono comporre la tabella:

Quando confermiamo, il programma inserisce una tabella che noi possiamo modificare inserendo dei dati, cambiando la posizione (ci si avvicina ai bordi dell'oggetto fintanto che il cursore prende la forma della freccia con quattro direzioni), variando le dimensioni; o si trascinano le maniglie di dimensionamento (i quadratini verdi ai lati dell'oggetto)

MARE	10
MONTAGNA	6
CITTA' D'ARTE	4
PARCHI GIOCHI	3
CROCIERA	1

oppure ci si avvicina lentamente ai segmenti di separazione delle righe o delle colonne fintanto che il cursore non assume la forma di due segmenti paralleli con due frecce nelle direzioni opposte ←||→ quindi si trascina per aumentare o diminuire la larghezza della colonna oppure l'altezza della riga.

Oppure cambiando lo sfondo delle diverse celle se non è gradito quello proposto da Impress; è sufficiente posizionarsi all'interno di una cella della tabella e automaticamente appare la barra degli strumenti Tabella

dalla quale si può intervenire con le modifiche desiderate. Colore Rosso 5

MARE	10
MONTAGNA	6
CITTA' D'ARTE	4
PARCHI GIOCHI	3
CROCIERA	1

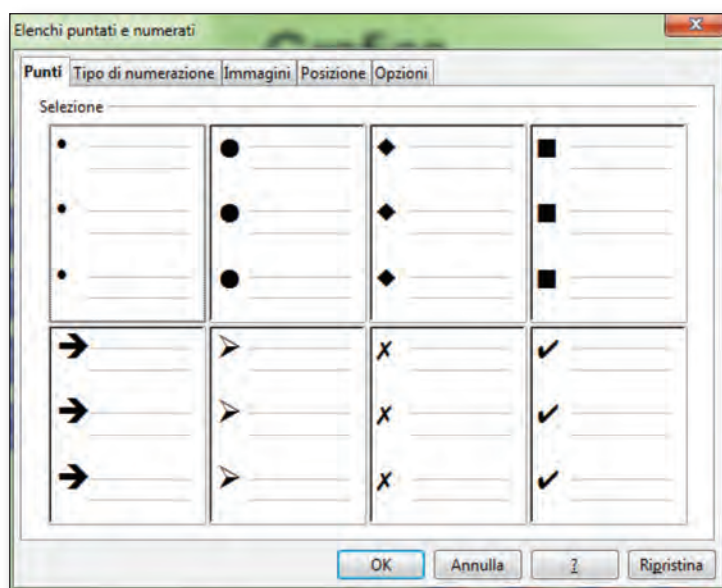
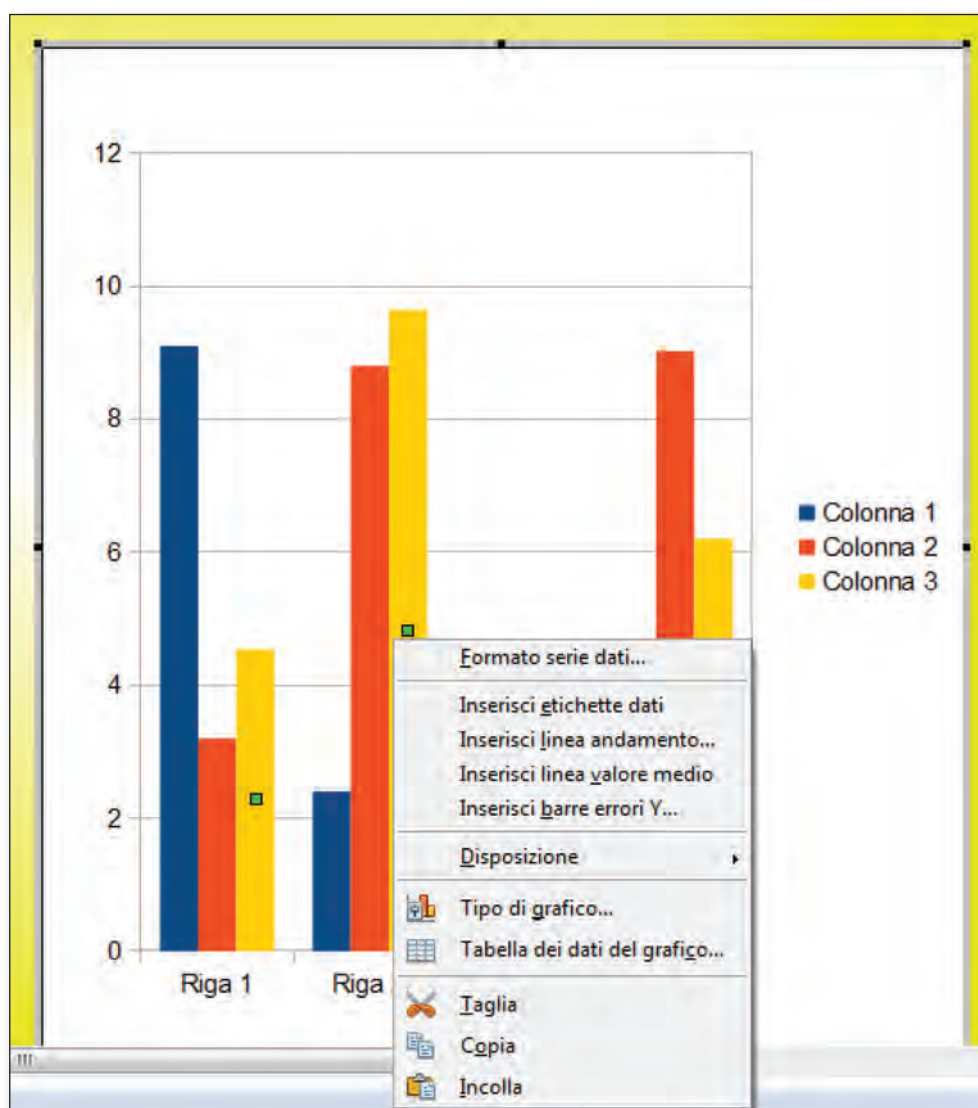
Tra le altre cose che è possibile fare con gli strumenti di questa barra ricordiamo:
variare le linee dei bordi delle singole celle ed il relativo colore,
inserire dei bordi alle celle, unire o separare delle celle,
allineare il testo dentro le celle in alto al centro o in basso,

aggiungere o eliminare delle righe o delle colonne .

Nell'esempio seguente è stata aggiunta una riga sopra alla prima, sono state unite le due celle, è stato scritto un titolo, sono stati inseriti dei bordi ed il testo è stato allineato al centro delle celle; infine la tabella è stata spostata più al centro.

I grafici

Adesso vogliamo inserire una slide che riassuma con un grafico i dati cha abbiamo inserito; dopo aver inserito una nuova "pagina" (secondo la definizione di Impress) con layout "Titolo, Contenuto2", inseriamo del testo nell'area di sinistra mentre nell'area di destra selezioniamo, dal quadrato centrale grigio , il quadrante che raffigura un grafico (quadrante in alto a destra). Nella slide compare un grafico (un istogramma) che non ha niente a che fare con i nostri dati poiché fa riferimento ad una tabella nascosta con dati arbitrari inseriti dallo stesso programma.

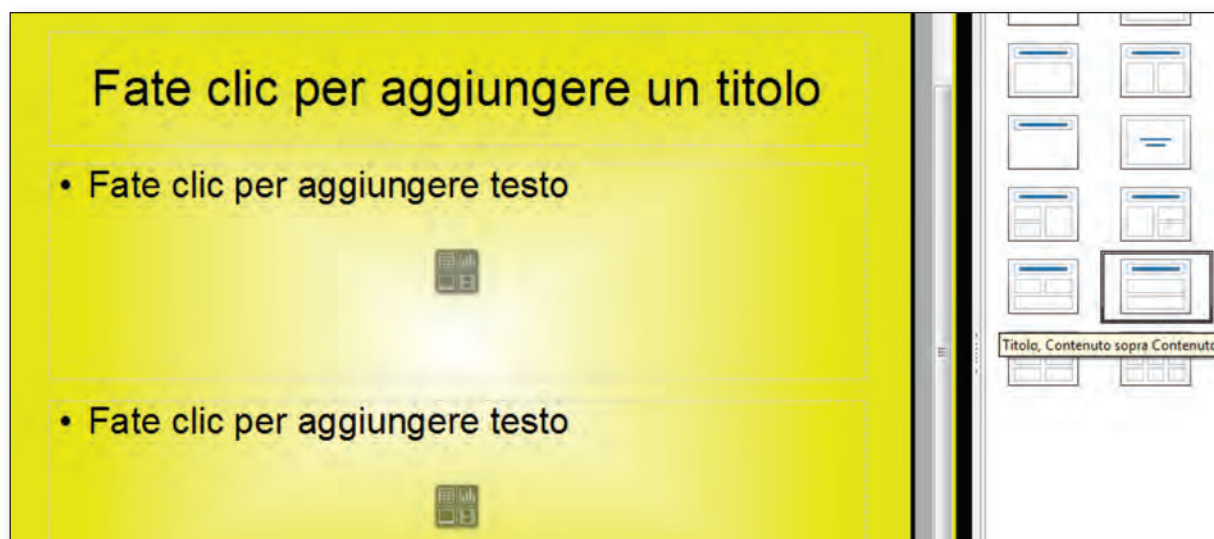
Osserviamo che automaticamente Impress attribuisce al testo una formattazione di elenco puntato; i punti elenco si possono modificare sezionando il testo e facendo un click col tasto desto sopra di esso, si apre un menù contestuale dal quale scegliamo la voce “Elenchi puntati e numerati …” quindi scegliamo tra i diversi punti disponibili, o immagini oppure possiamo farlo diventare un elenco numerato (anche qua con diverse opzioni).

E passiamo al grafico; esso è un istogramma che rappresenta dei dati che non hanno attinenza con la nostra indagine. Aggiorniamo la tabella dei dati facendo un doppio click sul grafico (in modo da aprire l’oggetto) e quindi un click destro. Si apre un menù contestale dal quale è possibile modificare sia il tipo di grafico (scegliere la voce “Tipo di grafico…”) che i dati (selezionare la voce “Tabella dei dati del grafico”).

Ho scelto il tipo di grafico a torta tridimensionale e per quanto riguarda i dati ho cancellato i dati inutili e modificato quelli necessari; è stata aggiunta anche una riga sotto con il pulsante .

	Categorie	Valori Y	Valori Y	Valori Y
1	mare	10		
2	montagna	6		
3	città d'arte	4		
4	parchi giochi	3		
5	crociera	1		

Il risultato finale è il seguente:

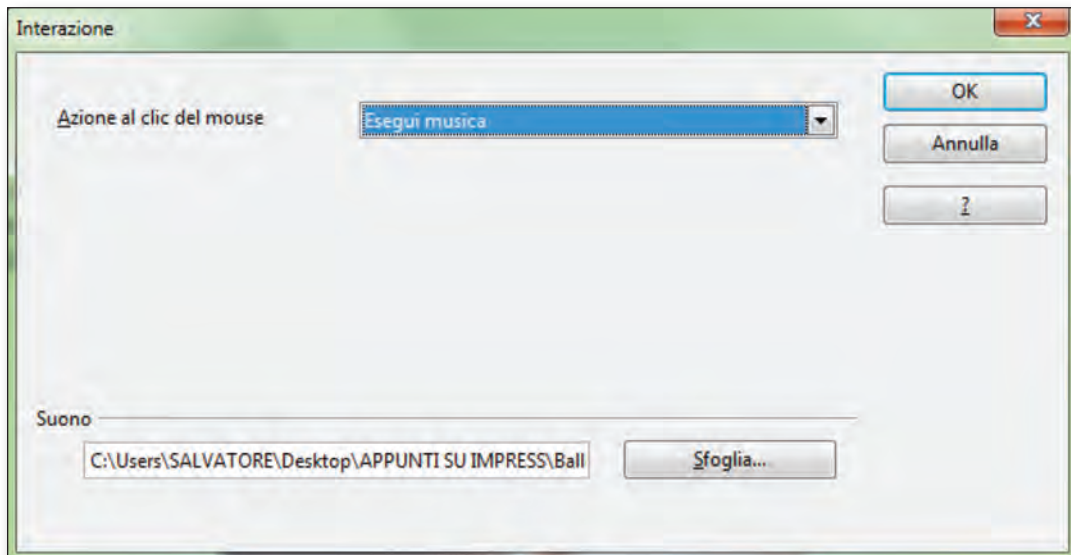
Dopo aver inserito una slide nuova con il pulsante pagina , dal pannello delle attività sulla destra, (cartella "Layout") è possibile scegliere come si deve presentare una diapositiva e quali devono essere i suoi contenuti. Da esso infatti possiamo scegliere in quante parti è divisa la slide e come si dispongono; in ognuna di esse si può scegliere di inserire un testo (di solito come elenco puntato) oppure altri oggetti.

Oggetti multimediali

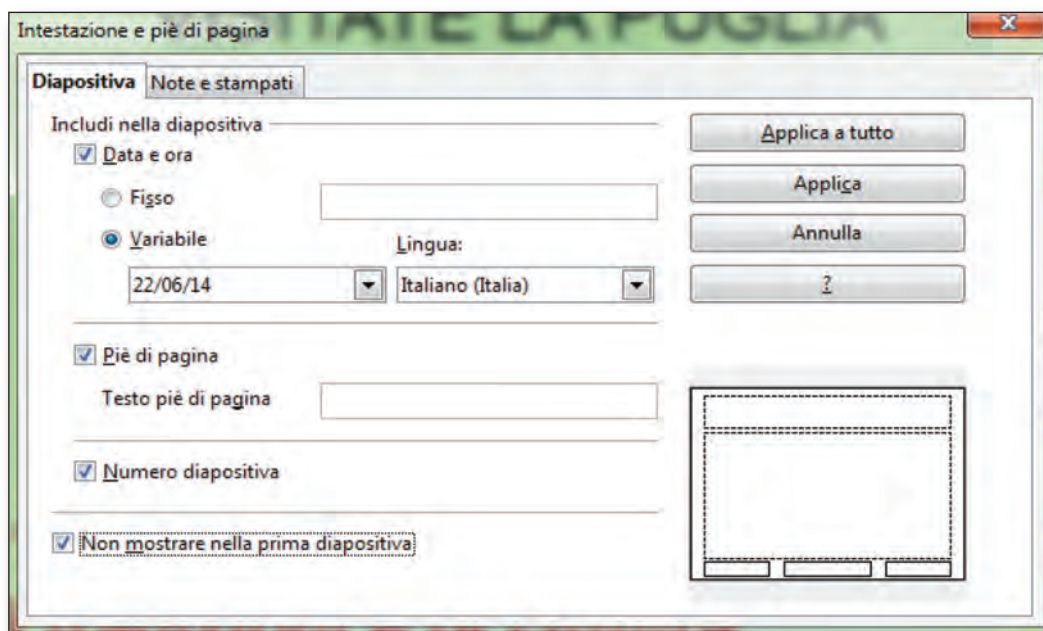
Ora creiamo una diapositiva che contenga del testo e un filmato e una diapositiva con del testo e un file musicale. Nel primo caso aggiungiamo la nuova slide e scegliamo il layout “Titolo, Contenuto sopra Contenuto”;

Inseriamo il titolo, del un messaggio promozionale nella zona inferiore e un video nella parte centrale. Nel messaggio promozionale della zona inferiore è stato cancellato il punto elenco che Impress genera automaticamente. L’inserimento del video e dell’audio è possibile cliccando nel quadrato centrale di ogni zona nel quadrante in basso a destra

Si apre una finestra dalla quale è possibile selezionare il percorso dal quale prelevare il video.

Le singole aree sono ridimensionabili e riposizionabili secondo il nostro gusto personale; ad esempio se vogliamo dare più enfasi all’area del video, la possiamo ingrandire e spostare al centro della slide aprendo il menù contestuale con un click destro e scegliendo la voce “Allineamento”.

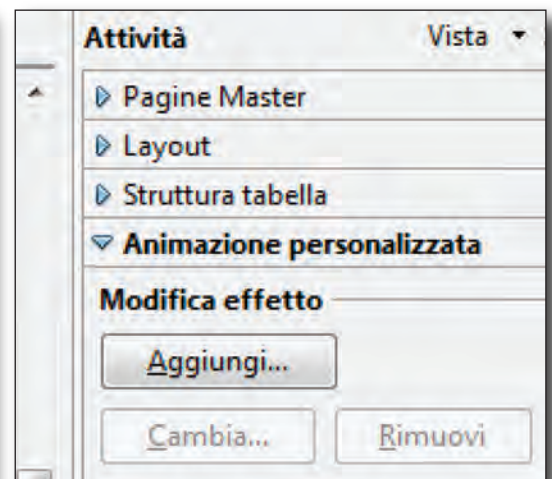
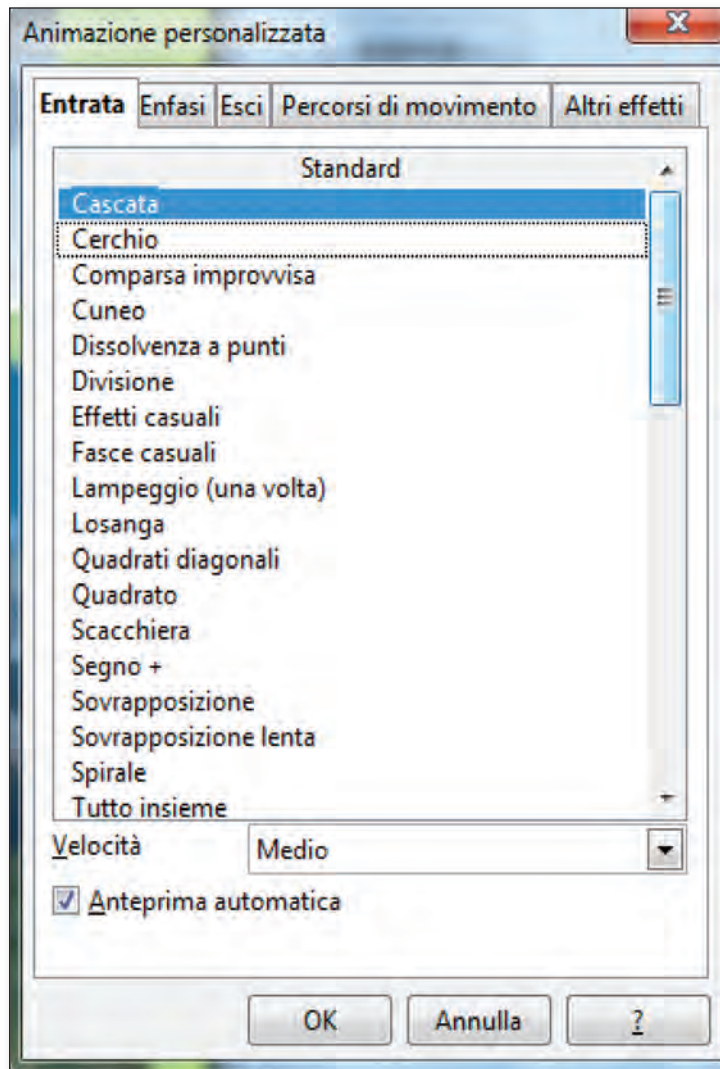
Nel caso della diapositiva con file audio scegliamo invece il layout “Titolo e Contenuto 2”; nella casella di sinistra inseriamo del testo e in quella di destra inseriamo il file audio; però non useremo lo stesso pulsante che abbiamo usato per inserire il video semplicemente perché durante l’esecuzione vedremo una casella nera che è antiestetica da vedere. Inseriremo invece una immagine mediante il pulsante “Da file” della barra “Disegno”. Facciamo click destro sull’immagine appena inserita e scegliamo la voce “Interazione” dal menù contestuale quindi nella scheda corrispondente selezioniamo l’azione “Esegui musica” e indichiamo il percorso dove prelevare il file sonoro.

Di tanto in tanto verifichiamo l’aspetto della nostra diapositiva o premendo dalla tastiera la funzione F5 oppure clicchiamo sul pulsante sulla barra degli strumenti.

Data e numero di pagina

Per inserire il numero di pagina e la data a tutte le diapositive selezioniamo la voce Barra dei Menù → Inserisci → Numero di pagina … (oppure Data e ora …); possiamo scegliere diverse impostazioni, per esempio che la data si aggiorni automaticamente a quella impostata nel computer al momento della presentazione e che il numero di pagina non sia inserito nella prima diapositiva:

Il risultato è questo:

Infine aggiungiamo un'ultima diapositiva di chiusura:

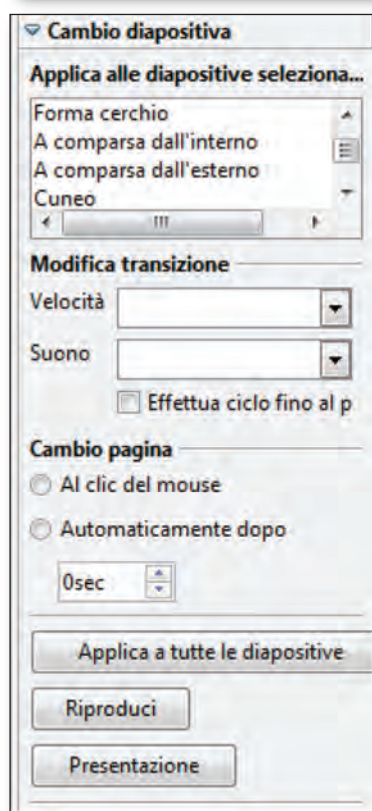
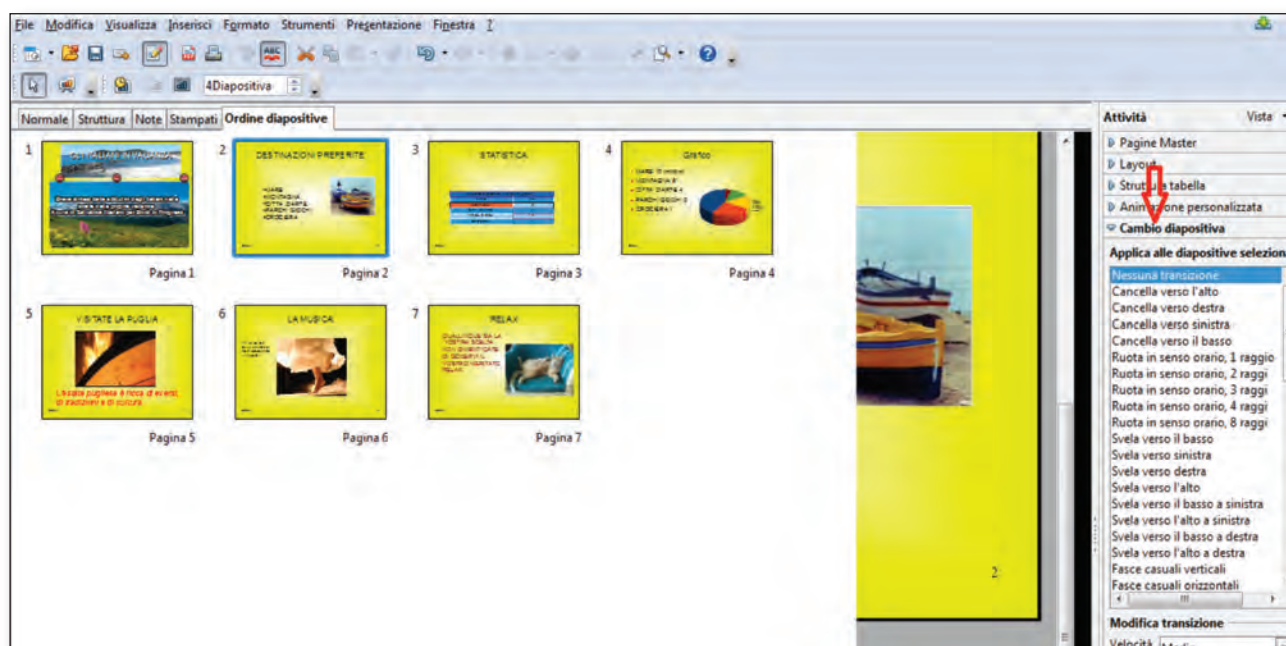
Animazioni e Transizioni

Quello che abbiamo fatto finora non differisce dal risultato che si poteva ottenere con un tradizionale proiettore di lucidi o di diapositive. La potenza del programma di presentazioni si apprezza quando rendiamo la nostra presentazione molto dinamica applicando animazioni e transizioni.

Le animazioni si possono applicare ai singoli oggetti di ogni diapositiva e permettono vari tipi di ingresso o di uscita dalla diapositiva durante la presentazione; le transizioni invece si applicano tra una diapositiva e l'altra per evitare che esista uno stacco netto nel passaggio tra le due slide. Il suggerimento è di non inserire molti effetti diversi nella stessa presentazione per evitare il rischio di stancare chi segue.

Per applicare una animazione ad un oggetto, è necessario innanzitutto selezionarlo, andare nel pannello delle attività (a destra dell'area di lavoro), selezionare la cartella "Animazione personalizzata", pulsante "Aggiungi" e scegliere una delle tante animazioni che Impress rende disponibili; bisogna provarle tutte per apprezzare quelle che più sono in linea con i nostri gusti. Da osservare che è possibile impostare la velocità di esecuzione dell'animazione in ingresso o in uscita, così è possibile anche impostare una traiettoria per un oggetto che entra o esce dalla diapositiva.

L'effetto associato ad un oggetto si può modificare o rimuovere sempre dallo stesso percorso; è possibile anche verificare immediatamente l'effetto applicato senza lanciare tutta la presentazione (Impress lo fa automaticamente) ma, grazie al pulsante "Riproduci" della stessa scheda, possiamo rivedere quante volte vogliamo l'animazione scelta finché non ne siamo pienamente convinti.

Nella presentazione di esempio allegata a questi appunti sono stati applicate diverse animazioni ai vari oggetti solo per provare i relativi effetti: se ne sconsiglia l'abuso soprattutto se la presentazione contiene molte slide.
Un'altra caratteristica importante per un programma di presentazioni è la possibilità di applicare degli effetti di transizione tra una diapositiva e l'altra. si può lavorare indipendentemente nella vista "Normale" oppure (meglio) nella vista "Ordine diapositive".

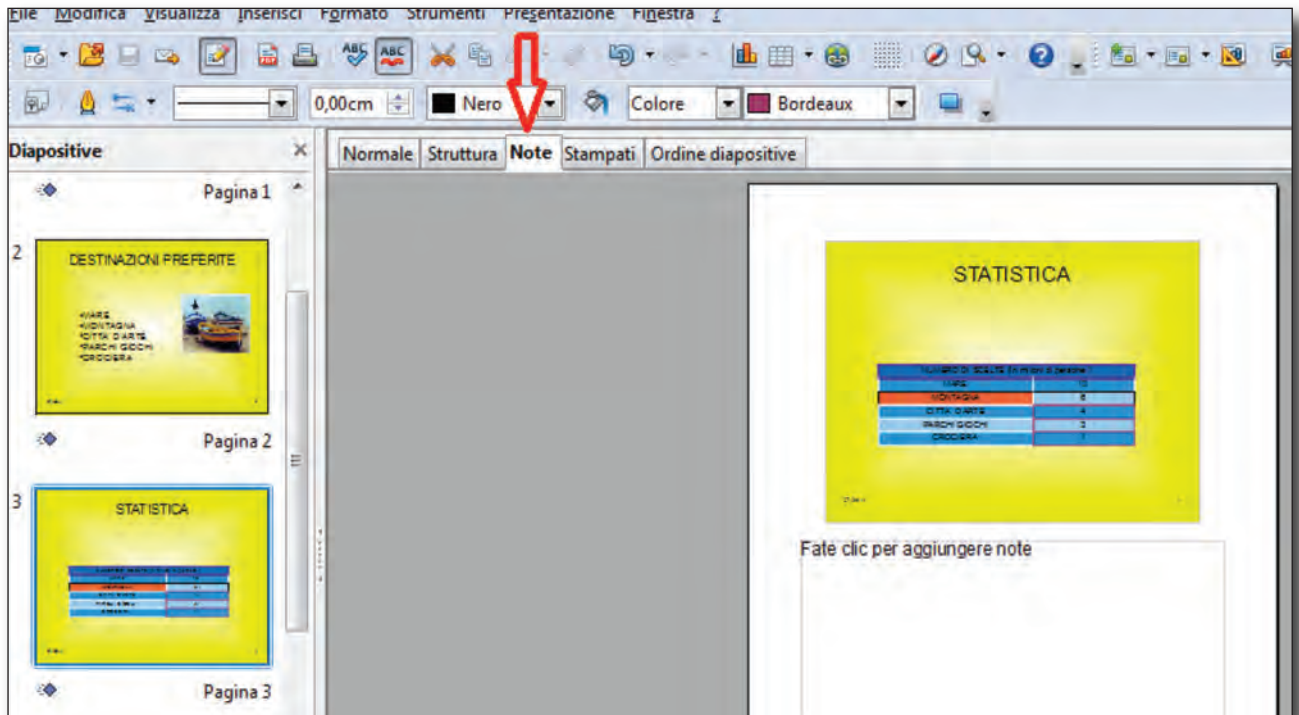
Se intendiamo applicare un effetto di transizione tra la prima e la seconda diapositiva (pagina) è necessario innanzitutto selezionare la seconda slide, quindi nel pannello delle attività a destra dello schermo selezioniamo la scheda "Cambio diapositiva" (di solito è in basso a destra).

È possibile applicare l'effetto di transizione scelto o solo alla diapositiva selezionata o a un gruppo di diapositive selezionate altrimenti a tutte le slide delle presentazione mediante l'apposito pulsante.

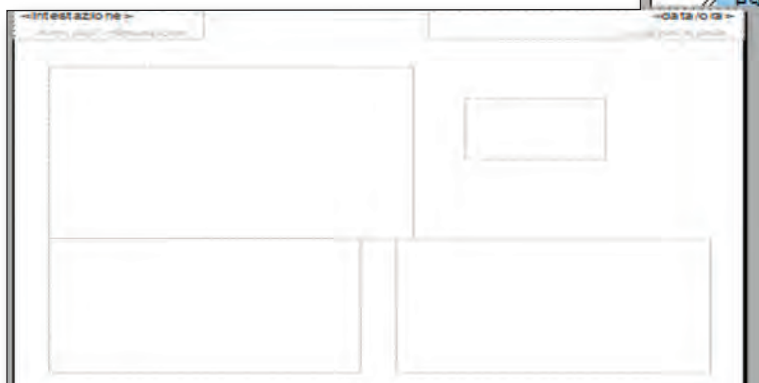
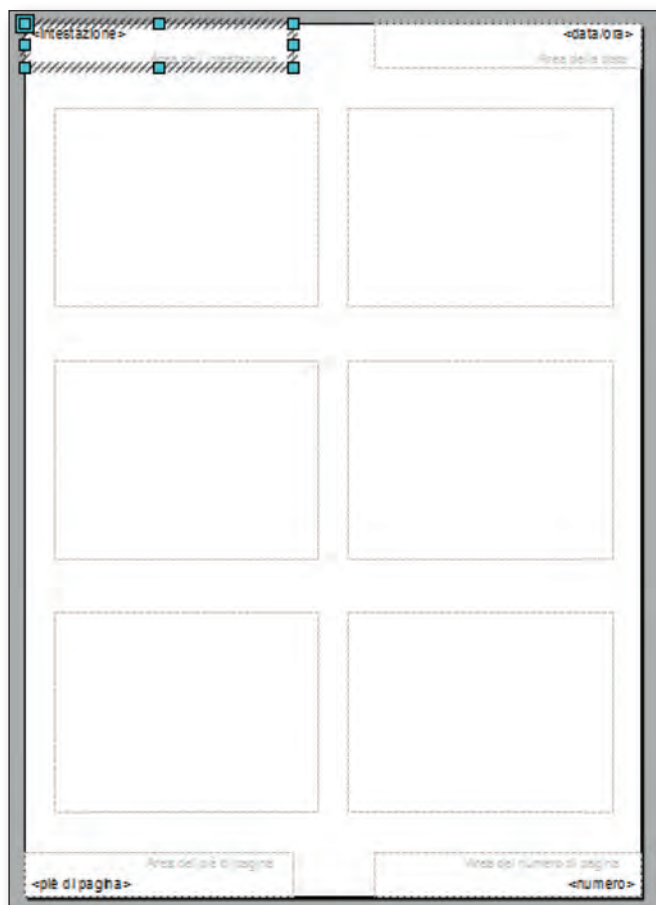
Sempre dalla stessa scheda è possibile impostare la velocità con la quale sarà eseguita la transizione ed eventualmente un suono tra quelli proposti da Impress. Da questa scheda è possibile impostare se il passaggio da una diapositiva all'altra deve avvenire mediante un click del mouse (o alla pressione di un qualsiasi tasto della tastiera) ovvero se deve avvenire automaticamente secondo degli intervalli di tempo predefiniti. In tal caso sarà necessario precisare dopo quanti secondi avviene il cambio di diapositiva; se l'effetto di transizione viene applicato a tutte le diapositive queste cambieranno sempre dopo l'intervallo di tempo indicato altrimenti, se vogliamo che i tempi siano differenti a seconda delle varie diapositive, doppiamo applicare l'effetto di transizione ed il relativo tempo, una diapositivaper volta.

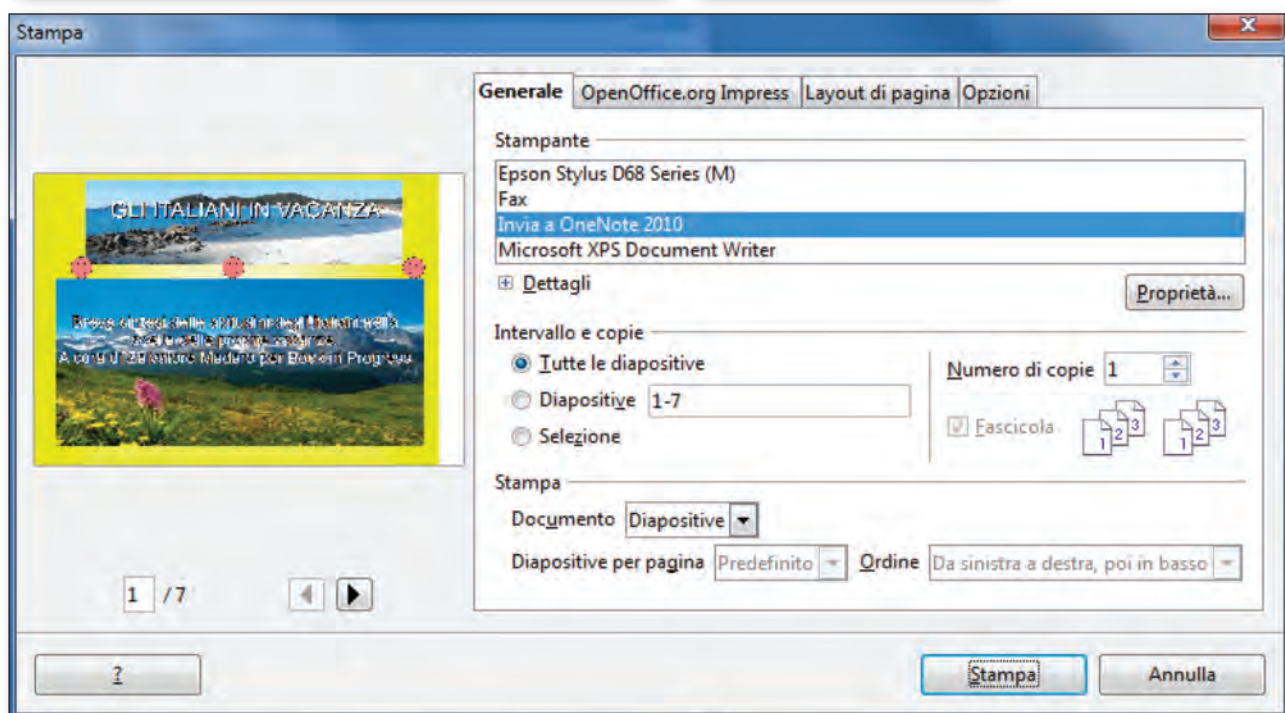
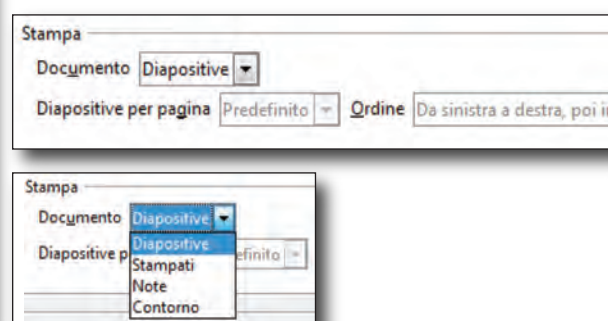
Il pulsante "Riproduci" permette di provare la transizione applicata tutte le volte che vogliamo mentre, se selezioniamo la prima diapositiva, il pulsante "Presentazione" avvia la presentazione dall'inizio. In alternativa possiamo premere il pulsante oppure premere il tasto F5.

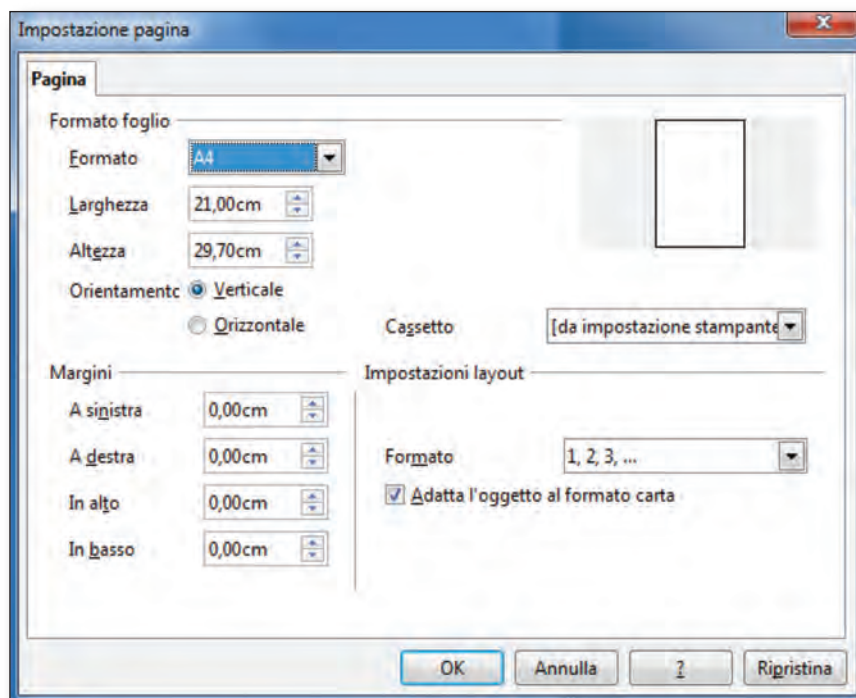
Sia nella vista "Normale" che nella vista "Ordine diapositive", quando applichiamo delle transizioni, sotto le icone delle diapositive, viene riportato il simbolo di un rombo

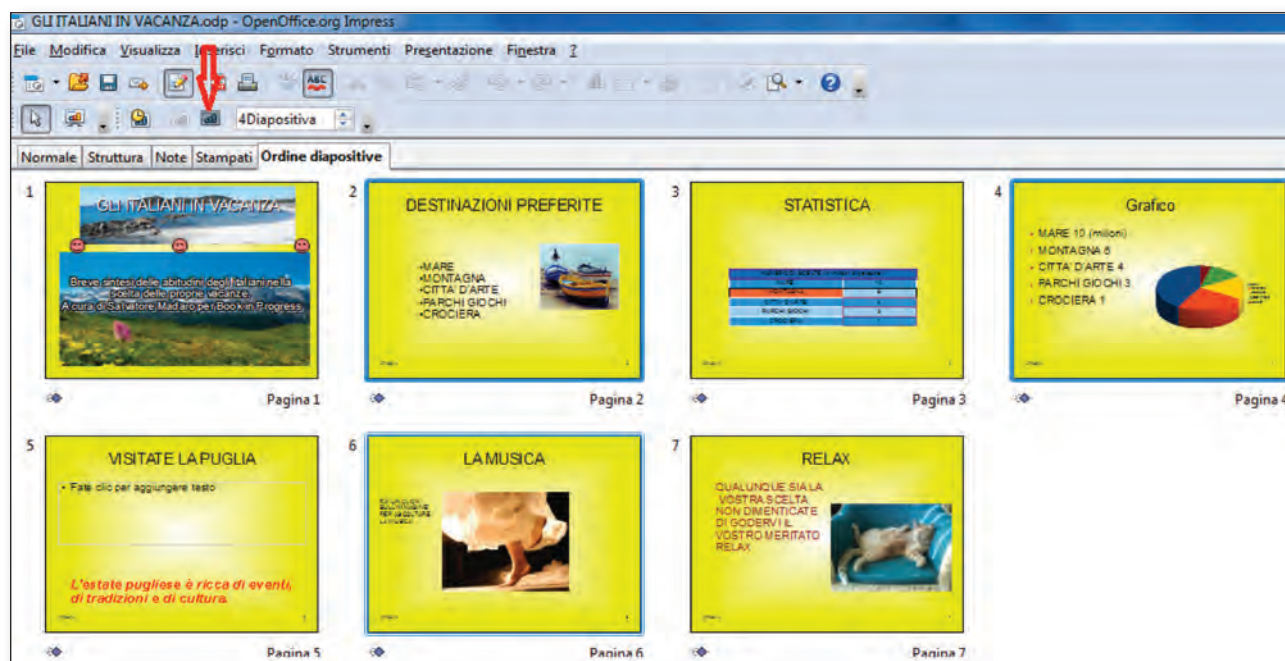
Note e Stampe

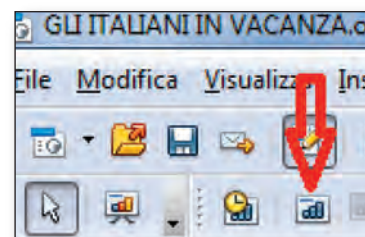
Ora che abbiamo realizzato la nostra presentazione, è possibile che si presenti la necessità di scrivere delle annotazioni, da tenere a portata di mano durante la presentazione, relativamente ai contenuti di qualche diapositiva. Per fare questo passiamo alla vista “NOTE” grazie alla linguetta della scheda presente sull’area di lavoro; in questo modo si apre la diapositiva selezionata e, al di sotto di essa, una casella di testo per inserire delle annotazioni.

È possibile stampare i promemoria inseriti nelle note in modo che siano da supporto al relatore durante la presentazione così come è possibile stampare le diverse diapositive in varie modalità.

La vista “STAMPATI” visualizza la stampa predefinita della presentazione con 6 diapositive per pagina, l’intestazione, il piè di pagina, il campo della data e l’ora di stampa che Impress preleva dall’orologio del PC, il numero di pagina. Ogni elemento dello stampato è selezionabile facendo click sul relativo bordo e modificabile a nostro piacere.

Quindi lo stampato predefinito è personalizzabile e, una volta salvata la presentazione, esso farà parte integrante di essa; per cui, premendo il pulsante della stampa diretta otteniamo in output questo stampato con il numero di diapositive e la disposizione che avremo impostato. In alternativa possiamo scegliere una stampa personalizzata dal seguente percorso:
Barra dei menù → File →Stampa.

Dalla scheda che si apre è possibile stabilire: quale stampante usare, se stampare tutte le diapositive o una parte di esse, quante copie del documento occorre stampare.
Nella parte inferiore della scheda si può impostare se stampare le diapositive (una slide per pagina), le note o gli stampati.

Nel caso si scelga la stampa di stampati, è possibile scegliere tra lo stampato predefinito oppure tra un certo numero di diapositive per pagina come anche l'ordine con il quale devono essere riprodotte nella stessa pagina più diapositive. La stampa delle note invece consente una slide per foglio con le relative note. La stessa scheda è possibile eseguire ulteriori personalizzazioni delle stampe e della modalità con la quale deve funzionare la stampante.

L'impostazione della pagina invece va eseguita seguendo il seguente percorso:
Barra dei menù → Formato → Pagina. Si apre una scheda dove si può impostare il tipo di foglio utilizzato, l'orientamento (verticale oppure orizzontale) e i margini.

Nascondere diapositive

Se la nostra presentazione si compone di numerose diapositive, è possibile che in alcuni casi sia necessario far vedere la presentazione per intero mentre in altri casi è opportuno che alcune diapositive siano saltate. Per evitare di farlo in modo approssimativo in presenza degli ascoltatori (perché è una causa di distrazione) si possono nascondere (non cancellare) le diapositive che non interessano in quel particolare contesto passando alla vista ORDINE DIAPOSITIVE, selezionando le diapositive da nascondere (Trascinando il mouse con il tasto sinistro premuto se le diapositive sono contigue oppure tenendo premuto il tasto Ctrl e facendo click sulle diapositive interessate) quindi si preme il pulsante "Nascondi diapositiva"

Le diapositive nascoste si riconoscono dal fatto che il numero di diapositiva viene scritto su un rettangolino grigio:

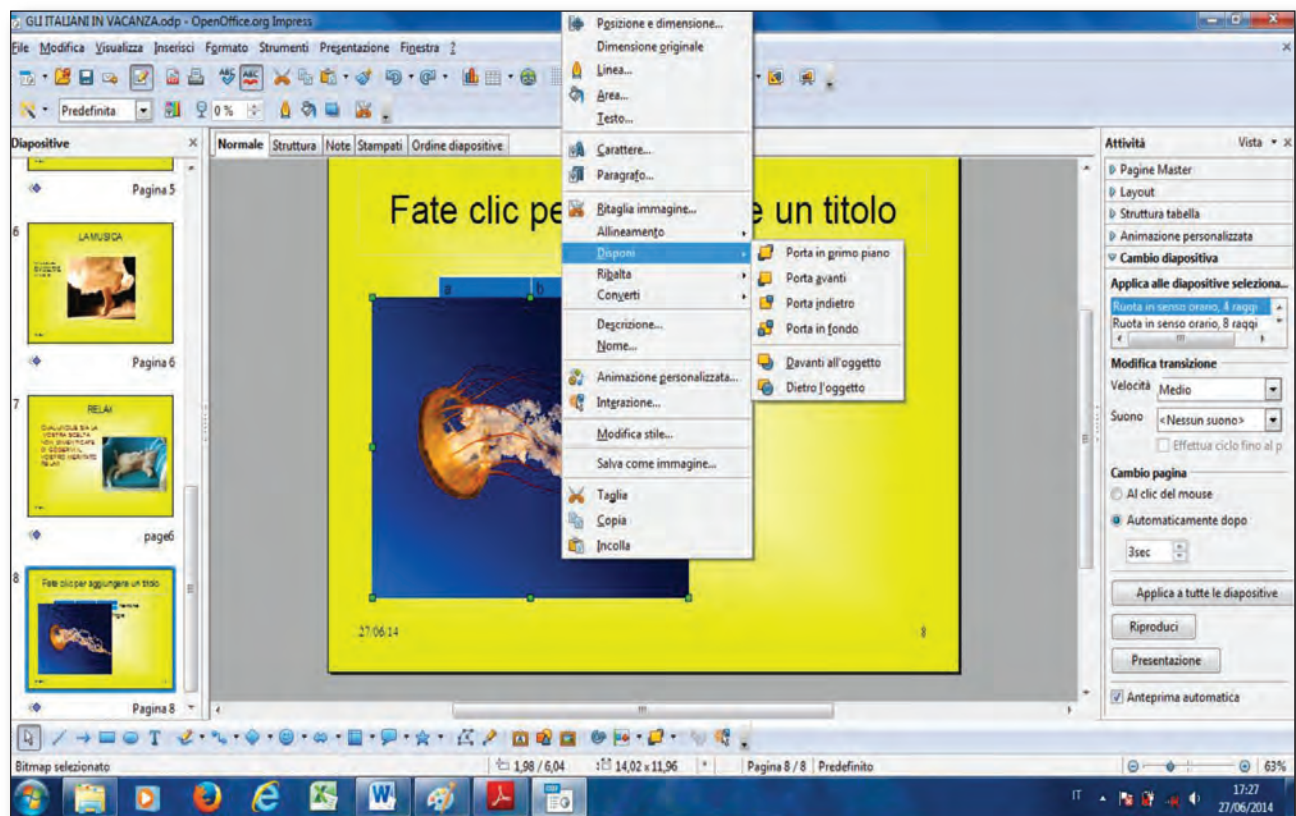
Per rendere nuovamente visibili le slide nascoste è necessario selezionarle e premere il pulsate "Mostra diapositiva " affianco a quello precedente e che ha la stessa icona.

Cambiare l'ordine della presentazione

È possibile che vogliamo cambiare l'ordine con il quale presentare le diapositive poiché non è detto che coincida con quello con il quale le abbiamo costruite. Per far questo ci posizioniamo nella vista "ORDINE DIAPOSITIVE", teniamo premuto il tasto sinistro del mouse mentre il cursore è sulla diapositiva da spostare (è possibile effettuare delle selezioni multiple), il cursore prende la forma di un segmento nero verticale, trascinare la diapositiva fino alla posizione desiderata e rilasciare; la diapositiva viene rinumerata automaticamente.

Gestione dei livelli in una diapositiva

Man mano che in una diapositiva inseriamo degli oggetti (testo, immagini, disegni, tabelle, ecc.) può accadere che si trovino sovrapposti uno sull'altro oppure vogliamo che lo siano ma in un determinato ordine scelto da noi; Impress sovrappone gli oggetti nell'ordine con il quale sono stati creati (ogni oggetto ha un livello di visualizzazione rispetto all'osservatore).

Per modificare l'ordine degli oggetti è sufficiente fare un click destro sull'oggetto, dal menù contestuale scegliere la voce "DISPONI" e quindi

quindi impostare una delle alternative previste:

Porta in primo piano: porta l'oggetto prima di tutti gli altri presenti nella diapositiva;
Porta avanti: l'oggetto avanza di un livello verso l'osservatore;
Porta indietro: l'oggetto si sposta indietro di un livello rispetto all'osservatore;
Porta in fondo: porta l'oggetto dietro a tutti gli altri presenti nella diapositiva;
Davanti all'oggetto: permette di spostare il livello di un oggetto prima di quello di un altro oggetto che verrà selezionato (dopo aver selezionato questa opzione, il cursore prende la forma di una mano con l'indice puntato che serve a selezionare l'altro oggetto di riferimento);
Dietro all'oggetto: permette di spostare il livello di un oggetto dopo quello di un altro oggetto che verrà selezionato (dopo aver selezionato questa opzione, il cursore prende la forma di una mano con l'indice puntato).

Un'altra opzione molto comoda è quello di associare più oggetti grafici come se si trattasse di uno soltanto. Per far questo è necessario selezionare tutti gli oggetto che intendiamo associare (click sul primo oggetto poi, tenendo premuto il tasto delle maiuscole, fare click sugli altri oggetti da raggruppare) click destro sul gruppo degli oggetti selezionati, dal menù contestuale scegliere la voce “RAGGRUPPA”.

L'operazione è reversibile, infatti se facciamo un click destro sul gruppo di oggetti raggruppati e scegliamo la voce “SEPARA” Separa ritorniamo ad avere i singoli oggetti separati.
Le immagini, i file audio e video presenti in questi appunti sono prese da Internet e non hanno alcun diritto di autore esplicitamente dichiarato.

10. CODING: dal Problema al Programma con Scratch e Python

Autori: **Domenico Deluso, Enrico Sartirana**
Rielaborazione a cura di **Angelo Oliva**

Introduzione: Pensiero computazionale e coding

E' ormai condiviso dalla comunità scientifica internazionale che l'informatica, oltre ad aver pervaso la nostra vita nei più reconditi risvolti, possa giocare un ruolo chiave in ambito formativo.
La formazione deve farsi carico di offrirci gli strumenti utili ad affrontare la vita di tutti i giorni, dotandoci delle competenze che ci permettano di diventare dei cittadini adulti, consapevoli ed in grado di operare delle scelte, per il nostro ed altrui benessere.

Viviamo in un mondo complesso, dove dobbiamo risolvere problemi complessi: l'informatica, da sempre, affronta problemi complessi studiando come raggiungere la soluzione e quali strategie adottare per mantenere minima la complessità.
Le strategie, i metodi, le modalità di affrontare problemi proprie dell'informatica sono racchiuse in quello che viene chiamato pensiero computazionale: uno studio teorico di questo argomento va oltre gli obiettivi di questo corso, però, ragionando da informatici, possiamo imparare questo stile.

C'è un percorso sicuro che ci permette di ragionare da informatici, ed è programmando (coding), scrivendo programmi: imparare a programmare è un lavoro lungo e complicato, ma esistono strumenti che ci permettono di scrivere codice e vedere funzionare il nostro programma in poco tempo.

L'obiettivo non è diventare programmatori, ma, forse divertendosi anche un po', raffinare il nostro modo di pensare imparando i concetti propri del pensiero computazionale, con la speranza di arricchire il nostro bagaglio di competenze per affrontare la nostra vita.

Di seguito proponiamo tre strumenti diversi e complementari per il coding: Scratch, Python e App Inventor.
Scratch permette di scrivere programmi attraverso un'interfaccia grafica, che semplifica il lavoro di imparare un vero e proprio linguaggio. Python, forse il più semplice linguaggio da imparare, permette di vedere all'opera in poche righe i nostri programmi. App Inventor è una piattaforma web di programmazione che consente di creare in modo molto semplice applicazioni (app), per smartphone , con sistema operativo Android.
In tutti e tre i casi, l'ambiente software ci permette di formalizzare il nostro ragionamento attraverso un programma, mentre lo strumento automatico (il computer) è in grado di eseguire il nostro ragionamento (il programma scritto) secondo le regole della logica: possiamo quindi osservare l'esecuzione del nostro ragionamento fuori dalla nostra mente.

I tre strumenti che imparerai
in questo e nel prossimo capitolo sono:

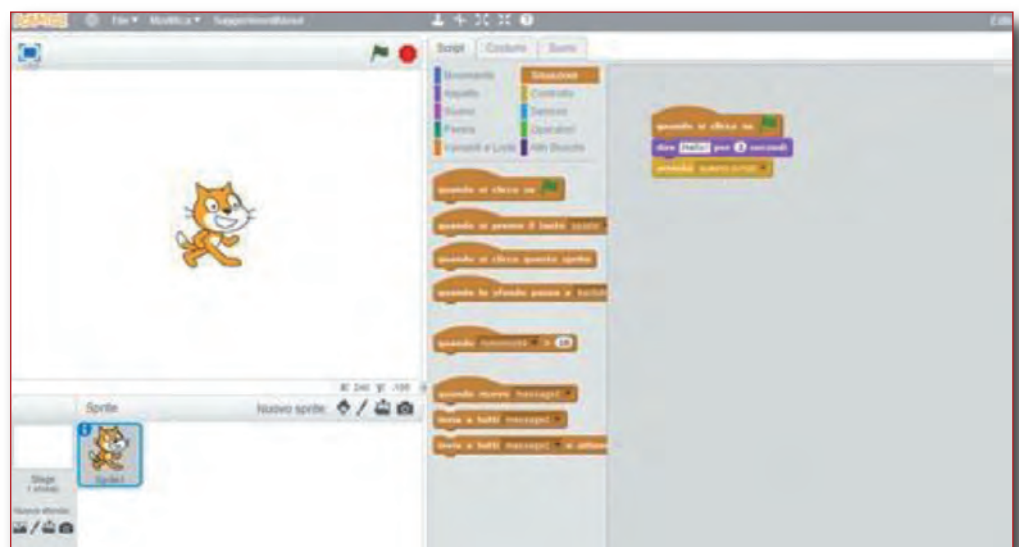
- **Scratch**
- **Python**
- **App Inventor**

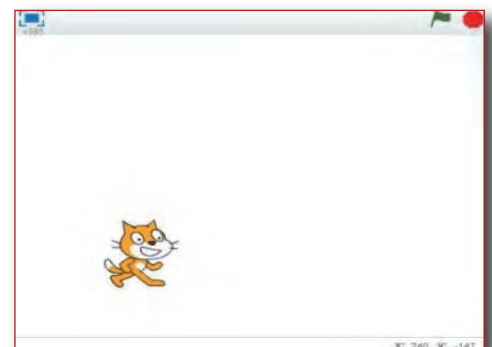
L'ambiente Scratch

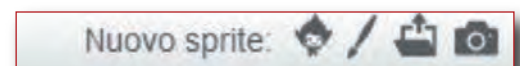
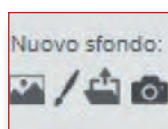
Scratch è un ambiente di programmazione visuale che consente di costruire programmi per risolvere problemi, produrre animazioni, simulazioni e interazioni tramite sequenze d'istruzioni eseguite da un computer. Questo software è il risultato di un progetto open source proposto dal Multimedia Lab del MIT di Boston e che vede tra i suoi finanziatori Google e Lego Foundation.

E' possibile effettuare il download di Scratch 2 versione offline, all'indirizzo: scratch.mit.edu/scratch2download/

Inoltre Scratch è utilizzabile anche online, basta iscriversi al sito. In tal caso si avranno a disposizione l'utilizzo di Scratch, la possibilità di usufruire di un vero e proprio cloud dei lavori e di condividere i propri file insieme a quelli della comunità degli "*scratchisti*".

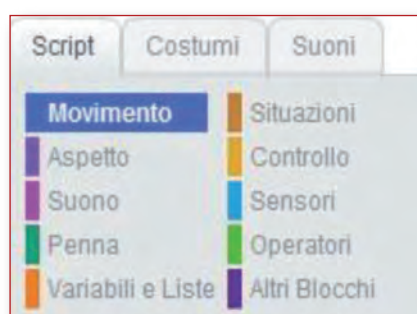
Dopo l'installazione e il lancio di Scratch l'ambiente chi si pone dinanzi all'utente è suddiviso in diverse sezioni che esaminiamo:

• *Stage*

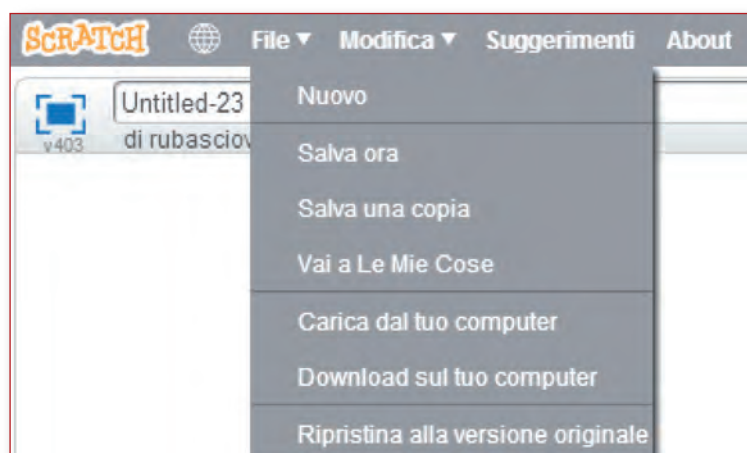
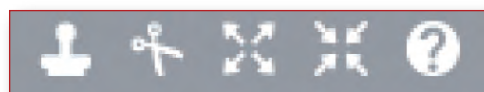
È l'area in cui è possibile osservare l'effetto derivante dall'esecuzione delle istruzioni di un programma. Il gattino che vedete nello stage si chiama sprite. Ogni sprite può essere animato o chiamato a svolgere il compito di interagire con l'utente.

Inoltre nello stage vi è da considerare lo sfondo in cui lo sprite è inserito, vi sono decine di sfondi messi a disposizione da Scratch, così come decine sono gli sprite.

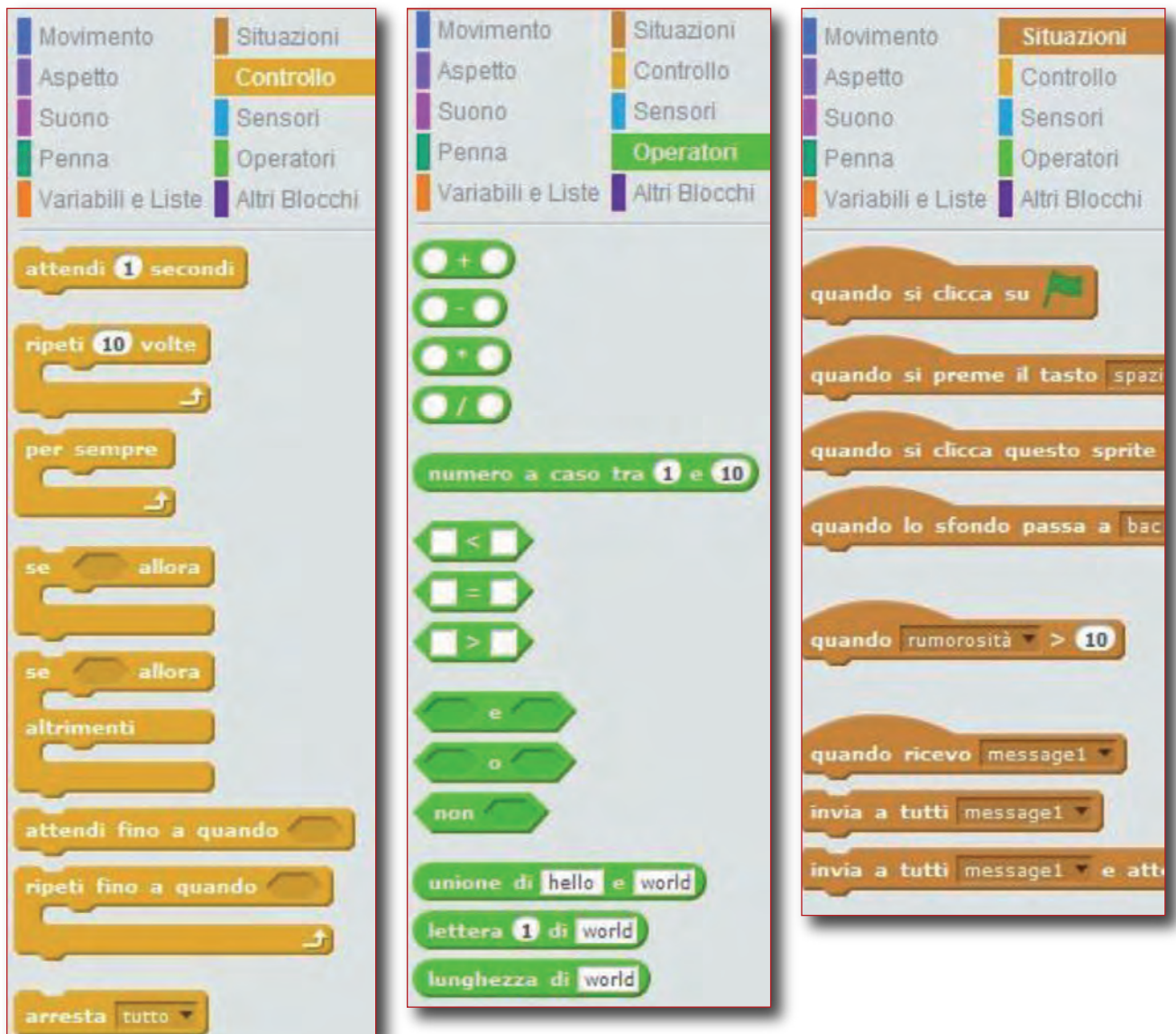
Per raggiungere e utilizzare tali risorse occorre cliccare su Nuovo Sprite

o su Nuovo Sfondo

Aree e icone sono riconoscibili sotto l'area sfondo.

• *Script*

Costituisce l'area che contiene le istruzioni del programma Ad ogni sprite e allo stage è associata un'area di script. Ogni istruzione appartiene ad un determinata categoria e viene trascinata dall'area categorie e quella degli script.

L'attività di programmazione avviene impilando una serie di oggetti grafici ciascuno dei quali costituisce un'istruzione, un comando che il computer deve eseguire all'atto dell'esecuzione. La sequenza di tali comandi-oggetto costituisce il programma. I comandi sono divisi in dieci differenti categorie, così come indicate sotto:

Scratch è dotato di una notevole libreria di sprite esplorabile attraverso i link dell„area *Nuovo sprite*, come pure di sfondi, che è possibile visionare tramite i link della *Nuovo sfondo*. Infine l'ambiente di programmazione consente l'ampliamento delle librerie, aggiungendo sprite e sfondi di propria creazione.

Con questi strumenti è possibile duplicare, eliminare, aumentare o diminuire la dimensione dello sprite o chiedere *aiuto*.

Inoltre nella stessa barra è presente anche quella dei menu, in particolare la voce *File* si dimostra utile per la relativa gestione dei programmi di volta in volta progettati e generati.

In precedenza abbiamo parlato delle categorie, accennando al fatto che queste si presentano come contenitori di istruzioni specializzate per essere utilizzate in determinate situazioni.

Ad esempio la categoria *Controllo* presenta istruzioni utili per effettuare controlli sui valori assunti dalle variabili consentendo all'algoritmo di ripetere sequenze di istruzioni o di decidere quale via intraprendere sulla base del risultato di un confronto. Gli *Operatori* vengono utilizzati invece per costruire condizioni ed espressioni aritmetiche. E ancora, le *Situazioni* rappresentano l'origine, l'inizio di una sequenza di istruzioni che potranno essere eseguiti al verificarsi degli eventi riportati da ciascun blocco.

• *Inizio/Fine*

Le due icone riportate sotto indicano la possibilità di dare avvio all'esecuzione delle istruzioni (*bandierina verde*) o di fermarne l'esecuzione (*pulsante rosso*).

Algoritmi

Un algoritmo è un insieme finito di istruzioni il cui scopo è risolvere un determinato problema.
L'esecuzione delle istruzioni di un algoritmo, provoca una serie di azioni in grado di trasformare i dati in ingresso in dati di uscita che rappresentano la soluzione del problema.

Ogni algoritmo comprende nella sua progettazione e scrittura tre fasi:
- input, per acquisire i dati
- elaborazione, per elaborare e trasformare i dati in ingresso;
- output, per visualizzare i risultati.

• *Input e Output*

È possibile utilizzare per la rappresentazione degli algoritmi quello che viene chiamato uno pseudo-linguaggio, cioè una rappresentazione dei comandi scritti in un linguaggio simile al linguaggio naturale (quello parlato dall'uomo) ma che non da origine ad ambiguità d'interpretazione, le istruzioni per acquisire i dati sono indicate come istruzioni di input. L'istruzione utilizzata è: Leggi (A). Mentre per visualizzare i dati, l'istruzione di output è data da Scrivi (B).

• *I costrutti*

Per realizzare algoritmi anche di elevata complessità è sufficiente disporre di tre strutture o costrutti fondamentali e ricombinarle tra loro secondo le esigenze:
- Sequenza. Le istruzioni sono poste una dopo l'altra, per cui la loro esecuzione avviene una dopo l'altra, raggiungendo la soluzione del problema.
- Selezione. Questo costrutto consente di progettare algoritmi che consento agli algoritmi di intraprendere vie alternative a seconda dei valori che assumeranno le variabili.
- Iterazione. Con tale costrutto è possibile eseguire ripetutamente una sequenza di istruzioni, per un numero finito di volte.

Inizio. Fine.
Ogni algoritmo presenta un inizio e una fine, questi due limiti vengono identificati nello pseudo-linguaggio dalle parole *Inizio e Fine*. Tutte le istruzioni che definiscono l'algoritmo sono poste tra questi due termini. Poiché ogni istruzione è non ambigua, la sua interpretazione sarà univoca.
Vediamo ora la formalizzazione degli schemi presentati.

• *Sequenza*

La sequenza viene rappresentata da una lista di istruzioni che vengono eseguite una dopo l'altra.

I_1
I_2
I_3
--
--
I_n

• *Selezione*

Se ***Condizione*** allora
  I_1
  I_2
  I_3
  --
  --
  I_n
altrimenti
  J_1
  J_2
  J_3
  --
  --
  J_m
FineSe

Lo schema di selezione consente all'algoritmo di intraprendere strade differenti, in base al valore di verità della Condizione. Seguendo lo schema sotto, quando la condizione è vera allora viene eseguita la sequenza $I_1, \dots, I_n$, altrimenti viene eseguita la sequenza $J_1, \dots, J_m$. Formalmente lo schema è descritto in questo modo:

• *Iterazione*

Ripeti Fino a che ***Condizione***
I_1
I_2
I_3
--
--
I_n
FineRipeti

Consente di ripetere una sequenza di istruzioni per un determinato numero di volte. La sequenza I_1, :….. , I_n viene eseguita finché la condizione non diventa vera. Lo schema è il seguente:

Osserviamo che in ogni schema può capitare di utilizzare uno qualsiasi degli schemi visti. Questo deve farci capire che in informatica non vi è la soluzione di un problema, ma possono co-esistere diverse soluzioni e che viene richiesta molta creatività, ma anche *scaltrezza* logico-matematica.

• *Condizioni*

Nei costrutti di selezione e ripetizione vengono utilizzate delle condizioni, queste non sono altro che predicati di carattere logico-matematico e possono assumere valori di verità vero o falso.
Esse utilizzano sia i simboli di relazione matematici =, >, <, >=, <= <>, che gli operatori logico-matematici *not*, *and*, *or.*
Se la condizione è costruita solo tramite i simboli relazionali: A<B; C>=B*4… la condizione si dice semplice.
Se la condizione utilizza anche gli operatori logico matematici allora la condizione si dice composta: (A>=5 and A<=7); (B<3 or B>9).

L'interpretazione della condizione composta deriva dal significato delle operazioni in logica matematica ed anche dalla priorità di esecuzione delle operazioni che è *not*, *and*, *or*:

Congiunzione

A	B	A and B
V	V	V
V	F	F
F	V	F
F	F	F

Disgiunzione

A	B	A or B
V	V	V
V	F	V
F	V	V
F	F	F

Negazione

A	Not A
V	F
F	V

• *Assegnamento*

Una particolare istruzione è quella di assegnamento, tramite cui è possibile assegnare un valore ad una variabile. La struttura tipica dell'assegnamento è:

Destinazione ← Sorgente

Dove la destinazione è sempre una variabile mentre la sorgente può essere un valore o un'altra variabile o un'espressione con valori e/o variabili, spesso invece del simbolo "←" viene utilizzato il simbolo "="
In generale allora un algoritmo può essere descritto da uno schema come questo:

A=5	Il valore 5 viene assegnato alla variabile identificata con il nome A
A=B	Il valore contenuto nella variabile B viene assegnato alla variabile A
A=A+B	La somma dei valori delle variabili A e B vengono assegnati alla variabile A
B=B+1	Il valore della variabile B viene incrementato di una unità.

dove I_1, …, I_n, possono essere singole istruzioni o costrutti.

Inizio
I_1
I_2

I_n
Fine

L'informatica teorica è permeata di logica matematica. In particolare ci si potrebbe chiedere se i costrutti visti per costruire gli algoritmi siano sufficienti per poter descrivere e risolvere i problemi e appartenenti alla classe dei problemi risolvibili. La risposta è affermativa ed è dimostrata dal teorema di Bohm-Jacopini.
Wikipedia offre una enunciazione semplice del teorema:
http://it.wikipedia.org/wiki/Teorema_di_B%C3%B6hm-Jacopini

• Dall'algoritmo a Scratch

Gli schemi visti possono essere formalizzati in Scratch. Una tabella esemplificativa e/o di riferimento viene riportata di seguito:

Tipo di istruzione	Pseudo Sintassi	Istruzione Scratch
Assegnamento	A=B	porta A a B
	B=3*B	porta B a 3 * B
	B=B+ 1	cambia B di 1
Condizionale	Se Condizione allora I_1 I_2 ------ IN [altrimenti J_1 J_2 ----- J_M] FineSe	se A < B allora se allora altrimenti
Input	Leggi(A)	chiedi Inserire Base e attendi porta A a risposta
Output	Scrivi(B)	porta Area a Base * Altezza mostra la variabile A
Iterazione – Ripetizione Ciclo PostCondizionale	Ripeti fino a che Condizione I_1 I_2 I_3 ---- In FineRipeti	ripeti fino a quando
Blocco Inizio	Inizio	quando si clicca su
Blocco Fine	Fine	arresta questo script

• Un esempio

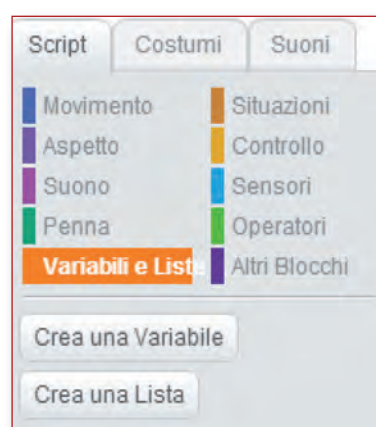
Scrivere un programma per calcolare l'area di un rettangolo conoscendo le misure della base e dell'altezza.

L'area è data dalla formula

*Area = Base * Altezza*

L'algoritmo è dato da:

Inizio
Leggi(Base)
Leggi(Altezza)
Area=Base * Altezza
Scrivi(Area)
Fine

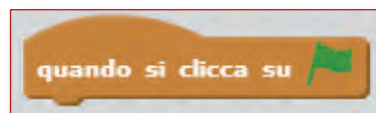
Definiamo le tre variabili utilizzando la categoria *Variabili e Liste* cliccare su Crea una Variabile

e dare un nome alla variabile da creare, in questo caso si sta definendo la variabile di nome Area. Premere su OK.

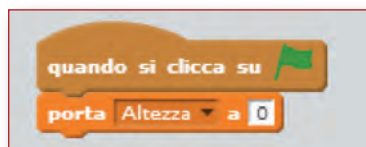
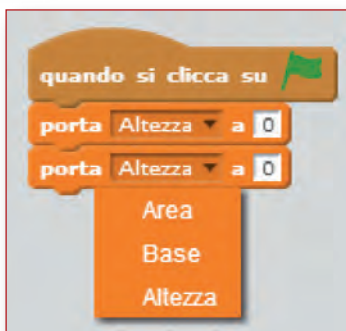
Nuova Variabile
Nome della variabile: Area
Per tutti gli sprite Solo per questo sprite
Cloud variabile (salvato sul server)
OK Annulla

Premendo su OK. La variabile è definita e inoltre vengono definite anche una serie di operazioni che è possibile effettuare su esse.

Continuate e fate lo stesso per le variabili Base e Altezza. Otterrete la seguente situazione:

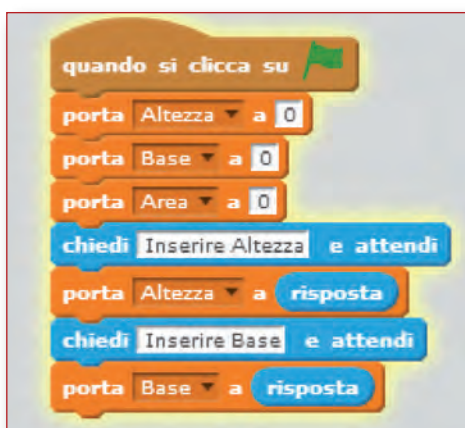
Mettendo i segni di spunta davanti alla variabile allora nello Stage saranno visibili le variabili ed il loro valore assunto:

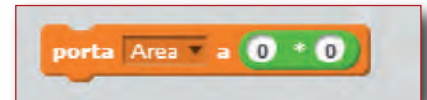
Cominciamo a scrivere la sequenza di istruzioni utili a risolvere il problema: dalla categoria *Situazioni*, clicca con il tasto sinistro del mouse su

e tenendo premuto il tasto trascina l'oggetto nell'area di script.

Ora introduciamo le variabili nell'area di script:
Cliccare sulla categoria Variabili e Liste
Cliccare e trascinare nell'area script l'oggetto di inizializzazione della variabile Altezza e impilarla con lo script iniziale, così come in figura:

Lo stesso per Base e Area scegliendo gli identificatori delle altre variabili dal menu a tendina:

La situazione sarà ora:

Ora occorre assegnare il valore sia alla base che all'altezza, e lo facciamo utilizzando un blocco preso dalla categoria sensori, ottenendo:

dove il blocco Risposta deriva ancora dalla categoria Sensori.

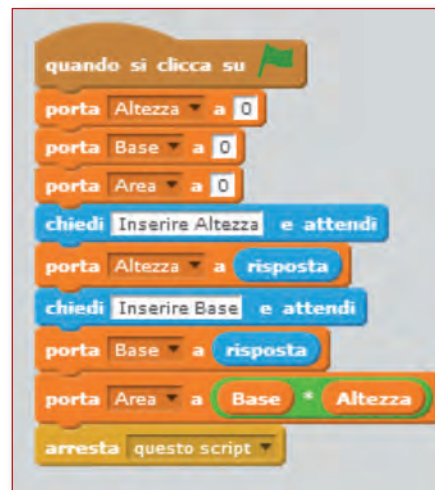
Analogamente per Base. Si otterrà:

Ora introduciamo l'assegnamento, rappresentato dalla formula per l'area, avremo bisogno dei seguenti oggetti:

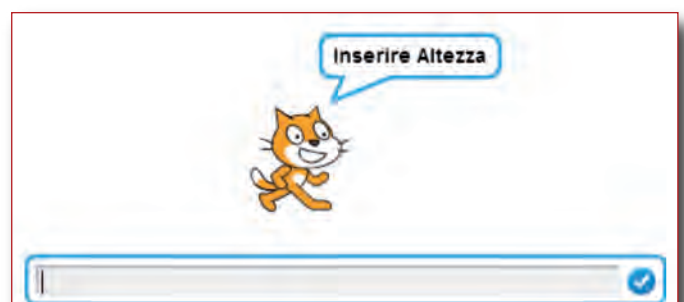
dove l'ultimo blocco è preso dalla categoria *Operatori*. L'espressione aritmetica si ottiene prelevando l'operazione di moltiplicazione da *Operatori* e adagiandola sul quadratino vuoto:

a loro volta, le variabili vengono adagiate sugli zeri, ottenendo la seguente situazione:

Quindi introduciamo la formula ottenendo:

```
quando si clicca su
porta Altezza a 0
porta Base a 0
porta Area a 0
chiedi Inserire Altezza e attendi
porta Altezza a risposta
chiedi Inserire Base e attendi
porta Base a risposta
porta Area a Base * Altezza
```

Quindi inseriamo lo script di chiusura prelevato dalla categoria Controllo:

```
quando si clicca su
porta Altezza a 0
porta Base a 0
porta Area a 0
chiedi Inserire Altezza e attendi
porta Altezza a risposta
chiedi Inserire Base e attendi
porta Base a risposta
porta Area a Base * Altezza
arresta questo script
```

Per l'esecuzione del nostro script che determina l'area del rettangolo basta cliccare sulla banderuola verde posta sull'area dello stage:

Inserire il valore nella casella di testo, dove si è posizionato il cursore, quindi cliccare sul pulsante di "spunta". Fare lo stesso per introdurre il valore dell'altezza.
Il valore dell'area si rende visibile nel box assegnato all'area.

• Schema di lavoro

La progettazione e costruzione di un algoritmo risolutivo di un problema e del relativo programma che lo rende eseguibile, passano attraverso diverse fasi che ne aiutano la progettazione rendendo la realizzazione indipendente dal linguaggio di programmazione adottato identificandolo quasi un semplice lavoro di traduzione. Queste fasi illustrate brevemente qui di seguito.

Fase 1: Analisi del problema; si cerca di capire attentamente cosa richiede il problema, quali sono i dati di input, output e le relazioni che legano gli uni agli altri e ancora se vi sono delle condizioni all'utilizzo di formule e infine se determinate sequenze di istruzioni e operazioni devono essere ripetute. Questa fase aiuta il lavoro che seguirà e in genere occupa gran parte del lavoro di progettazione.

Fase 2: Analisi dei dati e Schema di I/O. In questa fase vengono individuate e schematizzate le variabili di input e output, riportando in una tabella i nomi degli identificatori che si intendono dare alle variabili. Tali nomi devono essere significativi e mnemonici, perché in applicazioni complesse che utilizzano molti identificatori rendono più semplice l'attività di programmazione e aggiornamento del programma. Inoltre occorre per ciascun identificatore, individuare il formato (numerico, stringa, booleano, ecc) se è un dato di Input, Output, Lavoro (Work), e infine il significato del nome dell'identificatore all'interno del programma.

Fase 3: Algoritmo in pseudocodifica. Utilizzando le fasi precedenti e tenendo presente che l'algoritmo in generale si compone delle fasi di Input – Elaborazione – Output, si utilizzano le pseudoistruzioni per comporre l'algoritmo, comprendendolo tra i blocchi di inizio e fine.

Fase 4: Codifica, o realizzazione del Programma. L'algoritmo ottenuto viene tradotto istruzione per istruzione, nel linguaggio di programmazione adottato. Questa fase potrebbe richiedere qualche piccolo "artificio" pratico operativo che si impara per lo più frequentando il linguaggio e la disciplina.

Queste fasi di progettazione sono rese esplicite negli esempi che definiscono i capitoli relativi ai costrutti di sequenza, selezione, iterazione.

La sequenza

La sequenza è il primo e più semplice schema fondamentale della programmazione strutturata. Riportiamo qui tre esempi di questo schema, che si compone di sequenze di istruzioni, tra cui anche costrutti di selezione o iterazione che a loro volta potrebbero contenere altre sequenze, selezioni e ripetizioni… che possono ripetersi senza limite.
Tale schema è noto col termine ricorsione.

• Esempio 1

Dato il lato di un quadrato, progettare un'applicazione in grado di calcolarne perimetro e area.

Analisi del problema

Il problema ci chiede di progettare un'applicazione in grado di determinare perimetro e area di un quadrato conoscendone la misura del lato.
Il perimetro di un quadrato è dato dalla formula:

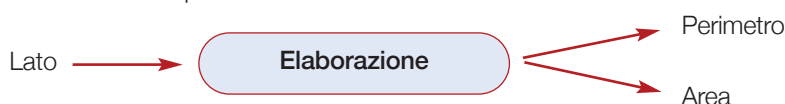
*Perimetro = 4*Lato*

Mentre per determinare l'area dello stesso utilizziamo la formula:

*Area = Lato*Lato*

Quindi in generale il nostro programma deve acquisire il lato, determinare il perimetro e l'area e infine visualizzare i risultati e/o le variabili ritenute necessarie.

Schema di I/O

L'algoritmo necessita di dati in input per poterli elaborare e quindi produrre degli output.
Possiamo schematizzare tale situazione in questo modo:

Analisi dei dati

Utilizzando le formule dell'analisi e lo schema possiamo individuare le caratteristiche e i nomi delle variabili utilizzate per risolvere il problema.

Identificatore	Formato	I/O/W	Significato
Lato	Numerico	I	Lato del quadrato
P	Numerico	O	Perimetro del quadrato
A	Numerico	O	Area del quadrato

I = input
O = Output
W = Work (Lavoro), sono variabili utilizzate per contenere risultati intermedi durante l'elaborazione.

Algoritmo di pseudocodifica

Individuiamo la sequenza di istruzioni che consente di acquisire i dati, elaborarli e visualizzarne i risultati.
L'algoritmo ha un blocco di inizio e fine. Le istruzioni saranno comprese tra questi due blocchi.

L'algoritmo deve acquisire il lato del quadrato, l'istruzione è:

Leggi (Lato)

Dopo l'acquisizione dei dati d'ingresso, si può passare alla fase di elaborazione utilizzando le formule:

P = 4*Lato

Area = Lato*Lato

Determinati perimetro e area, si può passare alla fase di output:

Scrivi (area)

L'algoritmo sarà quindi:

```
Inizio
    Leggi(Lato)
    P = 4*Lato
    Area=Lato*Lato
    Scrivi(P)
    Scrivi(A)
Fine
```

Codifica in Scratch

Nella figura riportata di seguito vi è la fase di codifica dell'algoritmo nel linguaggio proposto da Scratch, corredato di alcuni commenti esplicativi, in questo caso delle fasi di traduzione dalla pseudocodifica alla codifica Scratch, tenendo conto delle eccezioni da adottare (per esempio *nell'Inizializzazione*) che spesso dipendono dai linguaggi adottati.

> Tutti i linguaggi di programmazione prevedono la possibilità di inserire frasi di commento che non vengono riconosciute come istruzioni ma consentono al programmatore di inserire delle annotazioni utili durante l'attività di codifica.

Innanzitutto l'Inizio dell'algoritmo viene realizzato con il blocco evento **Quando**.
L'esecuzione comincia quando si clicca sulla bandierina verde.

La fase di input dell'algoritmo viene realizzata con l'istruzione Leggi. In questo modo il dato viene acquisito per le elaborazioni.

L'elaborazione degli stessi avviene attraverso le formule e relazioni. Si tenga presente che il significato del simbolo "=" nella nostra pseudocodifica è quello di assegnamento.

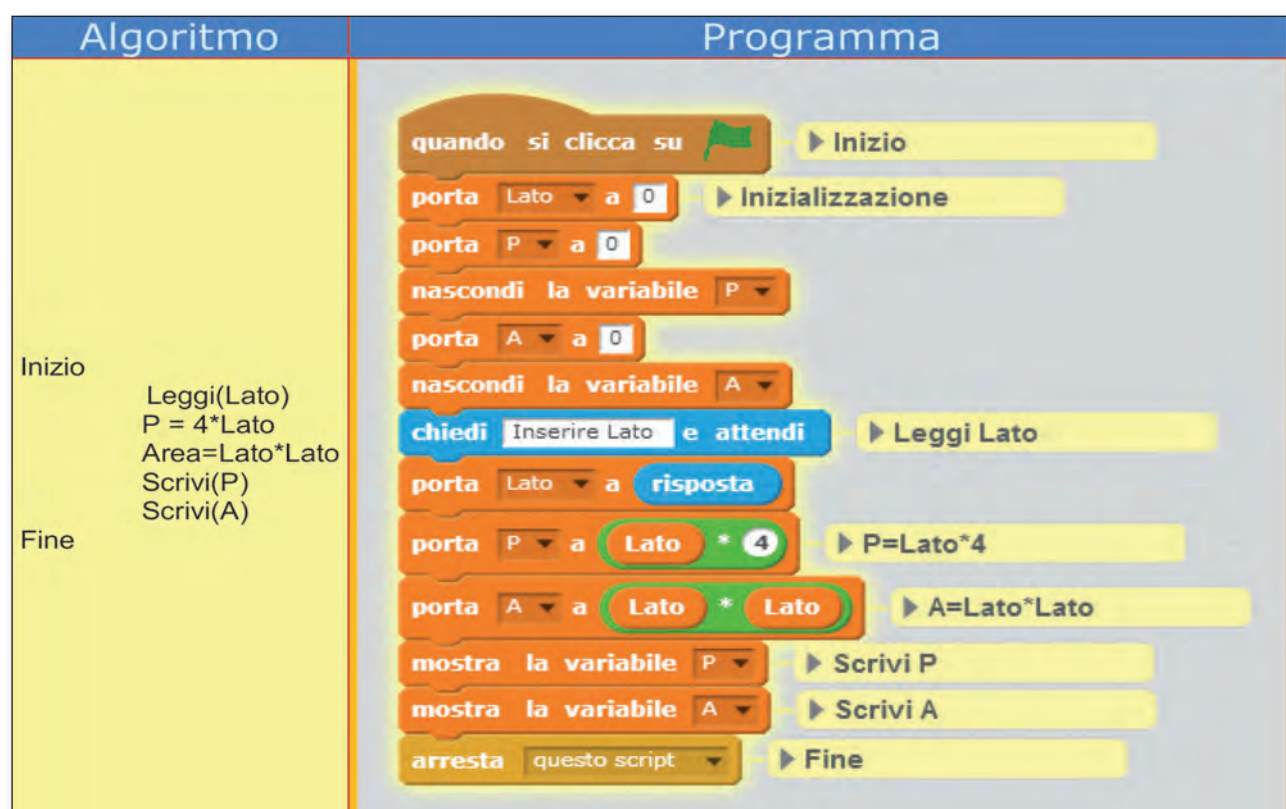
Ottenuti i risultati si può dar luogo alla fase di output, tramite l'istruzione Scrivi. In tal modo i risultati vengono visualizzati. La traduzione di ogni fase dell'algoritmo trova traduzione in Scratch, con qualche lieve modifica standard. Vediamo in che modo. Nel programma la fase di Inizializzazione viene effettuata perché, in caso contrario, se il programma venisse mandato in esecuzione una seconda volta, la sua esecuzione nella fase iniziale avrebbe come valori delle variabili i valori calcolati nella precedente esecuzione.

La pseudoistruzione Leggi(Lato) è realizzata interattivamente combinando in sequenza le istruzioni **Chiedi** e **Porta**. Dove Porta realizza l'assegnamento.

Analogamente con **Porta** vengono realizzati gli assegnamenti per determinare il perimetro e l'area.

La fase di output che nell'algoritmo viene esemplificata da Scrivi, in scratch lo realizziamo con il blocco **Mostra**.

Le label, etichette gialle che affiancano alcune istruzioni, rappresentano i commenti a istruzioni e fasi del programma. Esse aumentano la leggibilità dello stesso e aiutano a ricordarne e/o capirne la logica, all'autore, ma anche a programmatori esterni invitati ad analizzarli o a proseguirne il lavoro. È una buona tecnica di programmazione generalizzare le procedure costruite al fine di riutilizzarle nello stesso o in altri programmi, si parla in questo caso di riusabilità del codice, i commenti sono fondamentali per poter eseguire i necessari adattamenti.

L'istruzione Fine la si realizza con il blocco **Arresta** e l'opzione *questo script*.

• *Esempio 2*

Dati tre numeri, progettare un programma in grado di calcolarne la media aritmetica.

Analisi del problema

Il problema ci chiede di progettare un'applicazione in grado di determinare la media aritmetica tra tre numeri dati.

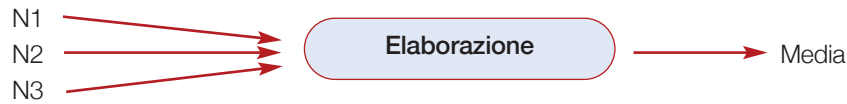
La media aritmetica tra n numeri è data dalla formula:

$$Media = \frac{a_1 + a_2 + ... + a_n}{n}$$

dove nel nostro caso n = 3.

L'algoritmo quindi deve acquisire i tre numeri, applicare la formula e visualizzarne il risultato, cioè la media.

Schema di I/O

L'algoritmo necessita di dati in input per poterli elaborare e quindi produrre degli output.
Possiamo schematizzare tale situazione in questo modo:

Analisi dei dati

Utilizzando le formule dell'analisi e lo schema possiamo individuare le caratteristiche e i nomi delle variabili utilizzate per risolvere il problema.

Identificatore	Formato	I/O/W	Significato
N1	Numerico	I	Primo numero
N2	Numerico	I	Secondo numero
N3	Numerico	I	Terzo numero
Media	Numerico	O	Media

I = input
O = Output
W = Work (Lavoro), sono variabili utilizzate per contenere risultati intermedi durante l'elaborazione.

Algoritmo

La fase di input viene realizzata con l'acquisizione dei tre numeri N1, N2, N3 utilizzando l'istruzione Leggi.

Quindi segue l'elaborazione che fa uso della formula individuata nella fase di analisi del problema. E infine l'output del risultato ottenuto con l'istruzione Scrivi.

```
Inizio
    Leggi N1
    Leggi N2
    Leggi N3
    Media=(n1 + n2 + n3)/3
    Scrivi Media
Fine
```

Codifica in Scratch

• *Esempio 3*

Effettuato un referendum si conoscono i voti a favore e quelli contrari. Progettare un'applicazione in grado da determinarne le relative percentuali.

Analisi del problema

Il problema ci chiede di progettare un'applicazione in grado di determinare le percentuali relative ai risultati di un certo referendum.
La percentuale dei voti favorevoli ai risultati si può determinare utilizzando la formula:

$$PercentialeSi = \frac{VotoSì * 100}{VotoSì + VotoNo}$$

dove VotoSì rappresenta i voti favorevoli, VotoNo i voti contrari.

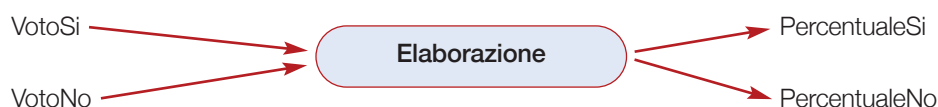
La percentuale dei voti contrari viene individuato dalla formula:

$$PercentialeNo = \frac{VotoNo * 100}{VotoSì + VotoNo}$$

o ancora tramite *PercentualeNo* = 100 - *PercentualeSì*

L'algoritmo deve acquisire i voti favorevoli e contrari, quindi applicare le formule per determinare le relative percentuali e visualizzare i risultati.

Schema di I/O

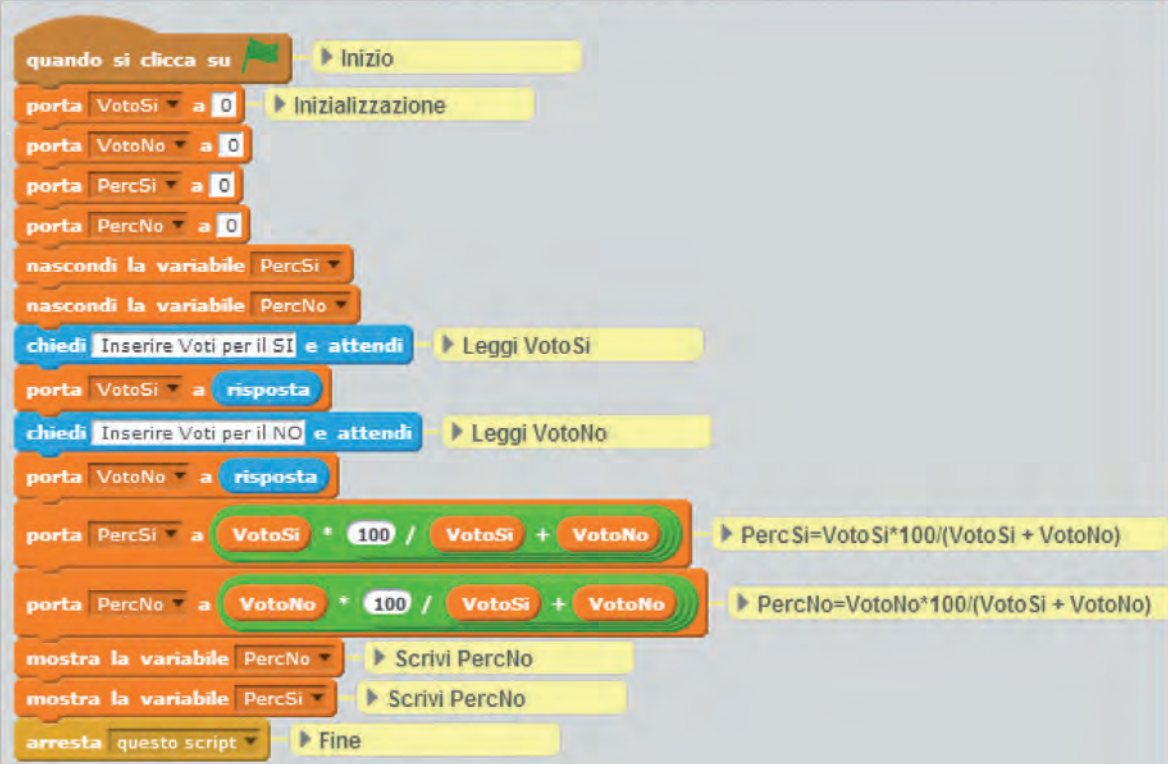
L'algoritmo necessita di dati in input per poterli elaborare e quindi produrre degli output.
Possiamo schematizzare tale situazione in questo modo:

Analisi dei dati

Utilizzando le formule dell'analisi e lo schema possiamo individuare le caratteristiche e i nomi delle variabili utilizzate per risolvere il problema.

Identificatore	Formato	I/O/W	Significato
Voto Sì	Numerico	I	Voti favorevoli
Voto No	Numerico	I	Voti contrari
PercSi	Numerico	O	Percentuale voti favorevoli
PerNo	Numerico	O	Percentuali voti contrari

Algoritmo di pseudocodifica

Vediamo ora l'algoritmo. Accanto alle istruzioni abbiamo aggiunto una frase preceduta da un apice " ' " per indicare che si tratta di un commento esplicativo. Tali frase durante l'esecuzione vengono ignorate.

```
Inizio
'Fase di input
        Leggi VotoSì        'acquisire voti favorevoli
        Leggi VotoNo        'acquisire voti contrari
'Fase di elaborazione
        PercSi = VotoSì *100/(VotoSì + VotoNo)        'calcolo percentuale favorevoli
        PercNo = VotoNo*100/(VotoSì + VotoNo)         'calcolo percentuale contrari
'Fase di output
        Scrivi PercNo        'output percentuale contrari
        Scrivi PercSì        'output percentuale favorevoli.
Fine Fine
```

Codifica in Scratch

Algoritmo

```
Inizio
  'Fase di input
  Leggi VotoSì
  Leggi VotoNo
  'elaborazione
  PercSì = VotoSì *100/(VotoSì + VotoNo)
  PercNo = VotoNo*100/(VotoSì + VotoNo)
  ' fase di output
  Scrivi PercNo
  Scrivi PercSì
Fine
```

Programma

```
quando si clicca su [bandiera]                         ▸ Inizio
porta VotoSi a 0                                       ▸ Inizializzazione
porta VotoNo a 0
porta PercSi a 0
porta PercNo a 0
nascondi la variabile PercSi
nascondi la variabile PercNo
chiedi Inserire Voti per il SI e attendi               ▸ Leggi VotoSi
porta VotoSi a risposta
chiedi Inserire Voti per il NO e attendi               ▸ Leggi VotoNo
porta VotoNo a risposta
porta PercSi a VotoSi * 100 / VotoSi + VotoNo          ▸ PercSi=VotoSi*100/(VotoSi + VotoNo)
porta PercNo a VotoNo * 100 / VotoSi + VotoNo          ▸ PercNo=VotoNo*100/(VotoSi + VotoNo)
mostra la variabile PercNo                             ▸ Scrivi PercNo
mostra la variabile PercSi                             ▸ Scrivi PercNo
arresta questo script                                  ▸ Fine
```

La selezione

Il costrutto di selezione consente all'algoritmo di intraprendere l'esecuzione di una sequenza di istruzioni anziché un'altra, sulla base del valore di verità di una condizione.

• *Esempio 1*

In un negozio se si acquista merce per più di 100 euro si ha diritto ad uno sconto del 20%. Progettare un'applicazione in grado di visualizzare la spesa scontata.

Analisi del problema

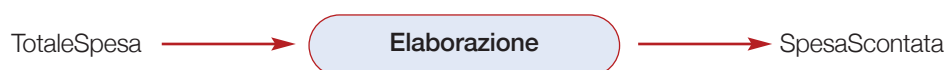
Dato un certo totale di spesa, se esso supera i 100 euro allora, per determinare la spesa scontata, utilizziamo la formula:

$$SpesaScontata = TotaleSpesa - (TotaleSpesa * \frac{20}{100})$$

Un'altra possibilità potrebbe essere:

$$SpesaScontata = TotaleSpesa * 0,80.$$

Nel caso in cui, invece, la spesa totale è inferiore o uguale ai 100 euro, l'algoritmo non deve fare niente.

Schema di I/O

Analisi dei dati

Identificatore	Formato	I/O/W	Significato
TotaleSpesa	Numerico	I	Spesa totale effettuata
SpesaScontata	Numerico	O	Spesa scontata del 20%

Algoritmo

Inizio
    Leggi TotaleSpesa
    Se TotaleSpesa>100 allora
        SpesaScontata=TotaleSpesa – (TotaleSpesa*20/100)
        Scrivi SpesaScontata
    FineSe
Fine

Codifica Scratch

```
quando si clicca su [bandiera]                         Inizio
porta TotaleSpesa a 0                                  Inizializzazio...
porta SpesaScontata a 0
nascondi la variabile SpesaScontata
chiedi Inserire Spesa Totale e attendi                 Leggi TotaleSpesa
porta TotaleSpesa a risposta
se TotaleSpesa > 100 allora
    porta SpesaScontata a TotaleSpesa - TotaleSpesa * 20 / 100     Calcola Spesa
    mostra la variabile SpesaScontata
arresta questo script                                  Fine
```

• Esempio 2

Ad un referendum è possibile votare con un Si o un No. Dati i risultati del referendum progettare un'applicazione in grado di visualizzare in ordine decrescente le percentuali dei risultati.

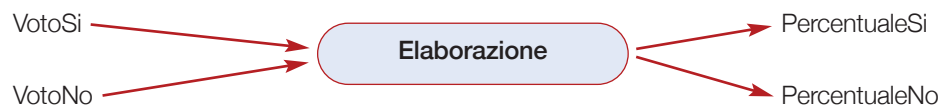
Analisi del problema

Il problema ci chiede di valutare i risultati in termini di percentuale, di un referendum, in cui è possibile votare con un Sì o un No. Dati quindi i risultati occorre determinare le percentuali e quindi visualizzare i risultati in ordine decrescente.
Il calcolo delle percentuali adottiamo le seguenti formule

*PercentualeSì = VotiSì *100/(VotiSì + VotiNo)*

PercentualeNo = 100 – PercentualeSì

Per determinare l'ordine con cui visualizzare i risultati occorre tener presente che le situazioni possono essere le seguenti:
❖ vincono i sì e allora si visualizza prima la percentuale dei sì quindi quella dei no
❖ vincono i no, infine il referendum si conclude in parità e allora l'algoritmo visualizza le percentuali di sì e no, che saranno uguali.

Schema di I/O

Analisi dei dati

Identificatore	Formato	I/O/W	Significato
Voto Sì	Numerico	I	Voti Si al referendum
Voto No	Numerico	I	Voti No al referendum
PercSi	Numerico	O	Percentuale voti per il Si
PerNo	Numerico	O	Percentuale voti per il No

Algoritmo di pseudo codifica

```
Inizio
'input
    leggi VotiSì
    leggi VotiNo
    'calcolo percentuali
    PSì=VotiSì*100/(VotiSì+VotiNo)
    PNo=100 – Psì
    se PSì > PNo allora 'vincono i Sì
        scrivi Psì
        scrivi PNo
    altrimenti
        se VotiSì<VotiNo allora     'vincono i No
                scrivi Pno
                Scrivi Psì
        altrimenti          'Parità
                scrivi PSì
                scrivi Pno
        finese
    finese
fine
```

Codifica Scratch

```
quando si clicca su [bandierina]                          ▸ Inizio
porta VotiSi a 0                                          ▸ Inzializzazione
porta VotiNo a 0
porta PSi a 0
porta PNo a 0
nascondi la variabile PSi
nascondi la variabile PNo
chiedi Inserire Voti per il SI e attendi                  ▸ Leggi Voti Si
porta VotiSi a risposta
chiedi Inserire Voti per il NO e attendi                  ▸ Leggi VotiNo
porta VotiNo a risposta
porta PSi a ((VotiSi * 100) / (VotiSi + VotiNo))          ▸ Calcola Percentuale dei Si
porta PNo a (100 - PSi)                                   ▸ Calcola Percentuale dei No
se VotiSi > VotiNo allora                                 ▸ VIncono i SI
    mostra la variabile PSi
    mostra la variabile PNo
altrimenti
    se PSi < PNo allora                                   ▸ Vincono i NO
        mostra la variabile PNo
        mostra la variabile PSi
    altrimenti                                            ▸ Parità
        mostra la variabile PNo
        mostra la variabile PSi
arresta questo script
```

• *Esempio 3*

Progettare un'applicazione che risolva le equazioni di primo grado.
In questo esempio utilizziamo lo sprite del "gattino" e il blocco "*dire*" per ottenere un effetto interattivo in fase di I/O.

Analisi del problema

Un'equazione di primo grado ha la forma:

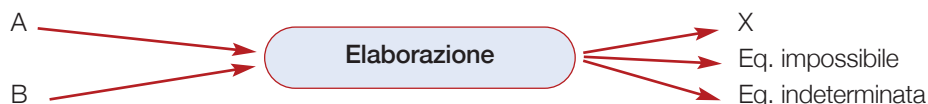
$$Ax = B$$

Se A ≠ 0 allora la soluzione dell'equazione è data da:

$$x = \frac{B}{A}$$

altrimenti se A=0 e B ≠ 0 allora l'equazione è impossibile, o ancora se A=0 e B=0 allora l'equazione è indeterminata.

L'algoritmo deve acquisire A e B, quindi controllare se A ≠ 0 può determinare la soluzione X, altrimenti se B≠0 allora l'equazione diventa impossibile, altrimenti se B = 0 allora l'equazione diviene indeterminata.

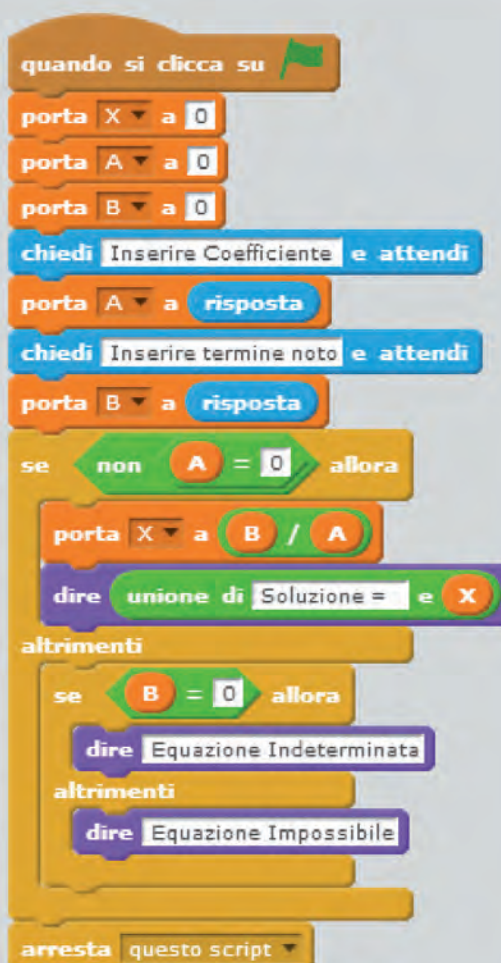
Schema di I/O

Analisi dei dati

Identificatore	Formato	I/O/W	Significato
A	Numerico	I	Coefficiente dell'incognita
B	Numerico	I	Termine noto
X	Numerico	O	Soluzione

Algoritmo di pseudo codifica

```
Inizio
    leggi A
    leggi B
    se non A=0 allora
        X=B/A
        scrivi X
    altrimenti
        se B=0 allora
                scrivi eq. indeterminata
        altrimenti
                scrivi eq. impossibile
        finese
    finese
fine
```

Codifica Scratch

```
quando si clicca su [bandiera]
porta X a 0
porta A a 0
porta B a 0
chiedi Inserire Coefficiente e attendi
porta A a risposta
chiedi Inserire termine noto e attendi
porta B a risposta
se non A = 0 allora
    porta X a B / A
    dire unione di Soluzione = e X
altrimenti
    se B = 0 allora
        dire Equazione Indeterminata
    altrimenti
        dire Equazione Impossibile
arresta questo script
```

L'Iterazione

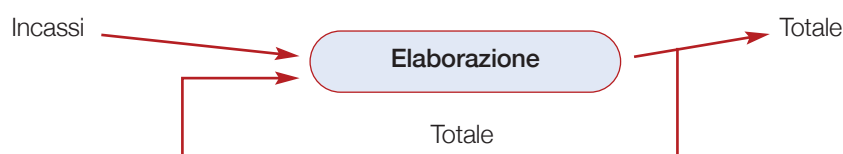
Questo costrutto consente di ripetere l'esecuzione di una sequenza di istruzioni per un certo numero di volte. Il numero di volte in cui la sequenza si ripete è dettato dalla cosiddetta *condizione di uscita* dal ciclo. Nella fattispecie Scratch, propone una prima forma di ciclo, detto post-condizionale, da cui si esce quando la condizione di uscita diventa vera. La sequenza cioè viene ripetuta finché la condizione di uscita non diventa vera. La sequenza di istruzioni da eseguire si dice *corpo del ciclo*.
In genere una variabile è preposta a portare il conto delle ripetizioni effettuate. Essa si dice *variabile contatore*. Nel corpo del ciclo è indispensabile che vi sia un'istruzione che modifichi i valori delle variabili coinvolte nella condizione di uscita in modo da farla diventare vera, altrimenti la condizione continuerà ad esser falsa e il ciclo non avrà mai fine, condizione di *loop infinito*, situazione contraria ad una delle proprietà fondamentali dell'algoritmo che consiste nel fatto che esso deve terminare in un tempo finito… Vediamo un primo esempio.

• Esempio 1

Gli incassi degli ultimi tre giorni di un esercizio commerciale, sono registrati su una tabella. Progettare un'applicazione in grado di determinare l'incasso totale.

Analisi del problema

Il problema ci chiede di determinare l'incasso totale di un esercizio commerciale, degli ultimi tre giorni. Quindi su una tabella sono registrati tre numeri, rappresentativi dei relativi incassi. Per determinare il totale si dovrebbero sommare i tre incassi, una sequenza ripetitiva che può essere risolta utilizzando il costrutto di ripetizione. Infatti, occorre ripetere per tre volte l'acquisizione di un incasso e l'aggiornamento del totale con l'assegnamento, in pratica la sequenza formalizzata sarà:
Leggi(Incasso)
Totale = Totale + Incasso
in questo modo al valore del totale viene sommato l' incasso appena acquisito, aggiornandolo.
Al termine della ripetizione la variabile Totale conterrà la somma dei tre incassi.

Schema di I/O

Analisi dei dati

Identificatore	Formato	I/O/W	Significato
Incasso	Numerico	I	Incasso
Totale	Numerico	O	Incasso Totale
K	Numerico	W	Contatore del ciclo

Algoritmo di pseudo codifica

Inizio
  Totale=0 'inizializzazione totalizzatore
  K=1 'inizializzazione contatore
  ripeti fino a che K > 3 'eseguire il corpo del ciclo finché k > 3 non diviene vera
  'inizio corpo del ciclo
    leggi incasso
    totale=totale+incasso
    k=k+1 'istruzione che modifica la variabile coinvolta nella condizione di uscita
  'fine corpo del ciclo
  fineripeti
  scrivi totale
fine

Analisi dell'algoritmo. Per questo primo esempio, studiamo un po' l'algoritmo. Esso ha una fase di inizializzazione del totalizzatore e del contatore. Il primo, prima di ogni suo aggiornamento vale 0. Il contatore invece vale 1. La condizione del ciclo k>3, finché è falsa porta all'esecuzione del corpo del ciclo. In particolare nel corso dell'esecuzione, l'incremento di k (k = k+1) porterà alla veridicità della condizione e all'uscita dal ciclo. Il corpo del ciclo presenta quindi le due istruzioni che consentono di acquisire l'incasso e aggiornare il totalizzatore con il valore dell'incasso appena letto.
All'uscita dal ciclo, nel totalizzatore sarà depositato l'incasso totale, per cui occorre visualizzarne il valore.
Un'ultima osservazione sul ciclo post-condizionale. Il corpo del ciclo può essere eseguito almeno una volta anche se la condizione è già vera, per cui occorre fare attenzione a questa caratteristica, poiché i valori di alcune variabili coinvolte nel corpo potrebbero essere modificate.

Codifica Scratch

```
quando si clicca su [bandiera]          ▸ Inizio
porta Totale a 0                        ▸ Inizializzazione Totale=0
nascondi la variabile Totale
porta Incasso a 0                       ▸ Incasso =0
porta K a 1                             ▸ Inizializzazione K=1
ripeti fino a quando K > 3              ▸ ripetere fino a che la condizione non diventa vera
    chiedi Inserire Incasso e attendi   ▸ Leggi Incasso
    porta Incasso a risposta
    porta Totale a Totale + Incasso     ▸ Aggiorna Totale=totale+Incasso
    cambia K di 1                       ▸ aggiorna contatore
mostra la variabile Totale              ▸ Scrivi Totale
arresta questo script                   ▸ Fine
```

• *Esempio 2*

La partecipazione ad un concorso si compone di cinque prove. Ciascuna con una sua valutazione compresa tra 1 e 10. Il concorso si può considerare superato se la media dei punteggi è almeno 7. Progettare un'applicazione in grado di determinare la media dei voti e l'esito del concorso.

Analisi del problema

Il calcolo della media di n numeri è dato da:

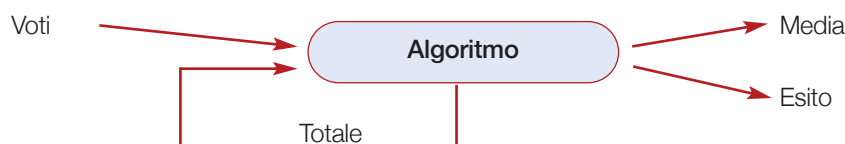
$$Media = \frac{a_1 + a_2 + \dots\dots + a_n}{n}$$

quindi occorre prima calcolare la somma totale degli n numeri e determinare la media aritmetica dividendolo per n.

Per determinare il totale, occorre costruire un algoritmo simile a quello dell'esempio precedente adottando un ciclo. In seguito basterà dividere tale totale per n ottenendo così la media aritmetica.
La condizione del ciclo è: k>5, mentre il corpo del ciclo è dato dalla sequenza che consente di acquisire il voto, aggiornare il totale, incrementare il contatore:

leggi voto
totale=totale+voto
k=k+1

all'uscita dal ciclo occorre determinare la media e infine verificare se il candidato ha superato il concorso controllando se la media sia almeno uguale a sette.
Nall'area Stage, lasciate che sia presente il *gattino*, in tal modo otterremo un qualche effetto interattivo.

Schema di I/O

Analisi dei dati

Identificatore	Formato	I/O/W	Significato
Voto	Numerico	I	Voto assegnato
Totale	Numerico	W	Totale voti
K	Numerico	W	Contatore ciclo

Algoritmo in pseudo codifica

```
Inizio
    totale=0
    k=1
    ripeti finchè k>5
        leggi voto
        totale=totale+voto
        k=k+1
    fineripeti
    media=totale/n
    se media>= 7 allora
        scrivi "Superato"
    altrimenti
        scrivi “Non Superato”
    finese
fine
```

Codifica Scratch

```
quando si clicca su [bandiera verde]
porta Totale a 0                         ▸ inizializzazioni
nascondi la variabile Totale
porta k a 1
porta Voto a 0
nascondi la variabile Voto
porta Media a 0
nascondi la variabile Media
ripeti fino a quando k > 5               ▸ ripeti finchè k>5
    chiedi Inserire Voto e attendi       ▸ leggi voto
    porta Voto a risposta
    porta Totale a Totale + Voto         ▸ aggiorna totale
    cambia k di 1                        ▸ incrementa k
porta Media a Totale / 5                 ▸ calcol media
mostra la variabile Media
se Media > 7 o Media = 7 allora          ▸ se media >=7 promosso
    dire PROMOSSO!
altrimenti                               ▸ altrimenti bocciato.
    dire NON PROMOSSO!!
arresta questo script
```

L'esito sarà visibile nell'area stage

• *Esempio 3*

In una tabella sono riportati, di volta in volta, i premi assicurativi riscossi da tre agenti da alcuni cliente. Man mano che i premi vengono riscossi, li si registra in una tabella che riporta il codice dell'agente e la somma riscossa. Ad un certo punto la tabella è formata dalle seguenti 10 righe.

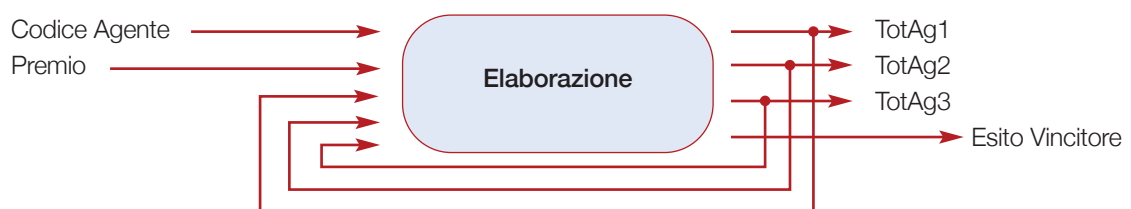
Agente	Premio
C01	50
C01	40
C03	50
C02	35
C02	40
C01	200
C03	100
C01	45
C03	100
C02	150

Progettare un'applicazione in grado di determinare, per ogni agente, il totale riscosso ed individuare l'agente che ha riscosso il totale più alto *(facendo l'ipotesi semplificativa che non vi siano situazioni di parità).*

Analisi del problema

Occorre scrivere un programma per
– determinare il totale dei premi riscossi da ciascun agente assicurativo;
– individuare chi ha il totale più alto.

Per il primo punto occorre percorrere la lista tramite un ciclo. Nel corpo del ciclo si acquisisce il codice dell'agente e la cifra riscossa, quindi in base al codice viene aggiornato il totalizzatore relativo;
all'uscita del ciclo si confrontano tra loro i totali per determinare il valore massimo.
La condizione di uscita dal ciclo viene individuata tramite un contatore ed è la seguente k>10, nel corpo viene inserita l'istruzione di incremento del ciclo: k=k+1.
Le inizializzazioni inseriscono i tre totalizzatori che partono da zero e il contatore che parte da uno.

Schema di I/O

Analisi dei dati

Identificatore	Formato	I/O/W	Significato
Agente	Carattere	I	Codice agente
Premio	Numerico	I	Premio assicurativo
TotAg1	Numerico	O	Totale premi agente C01
TotAg2	Numerico	O	Totale premi agente C02
TotAg3	Numerico	O	Totale premi agente C03

Algoritmo di pseudo codifica

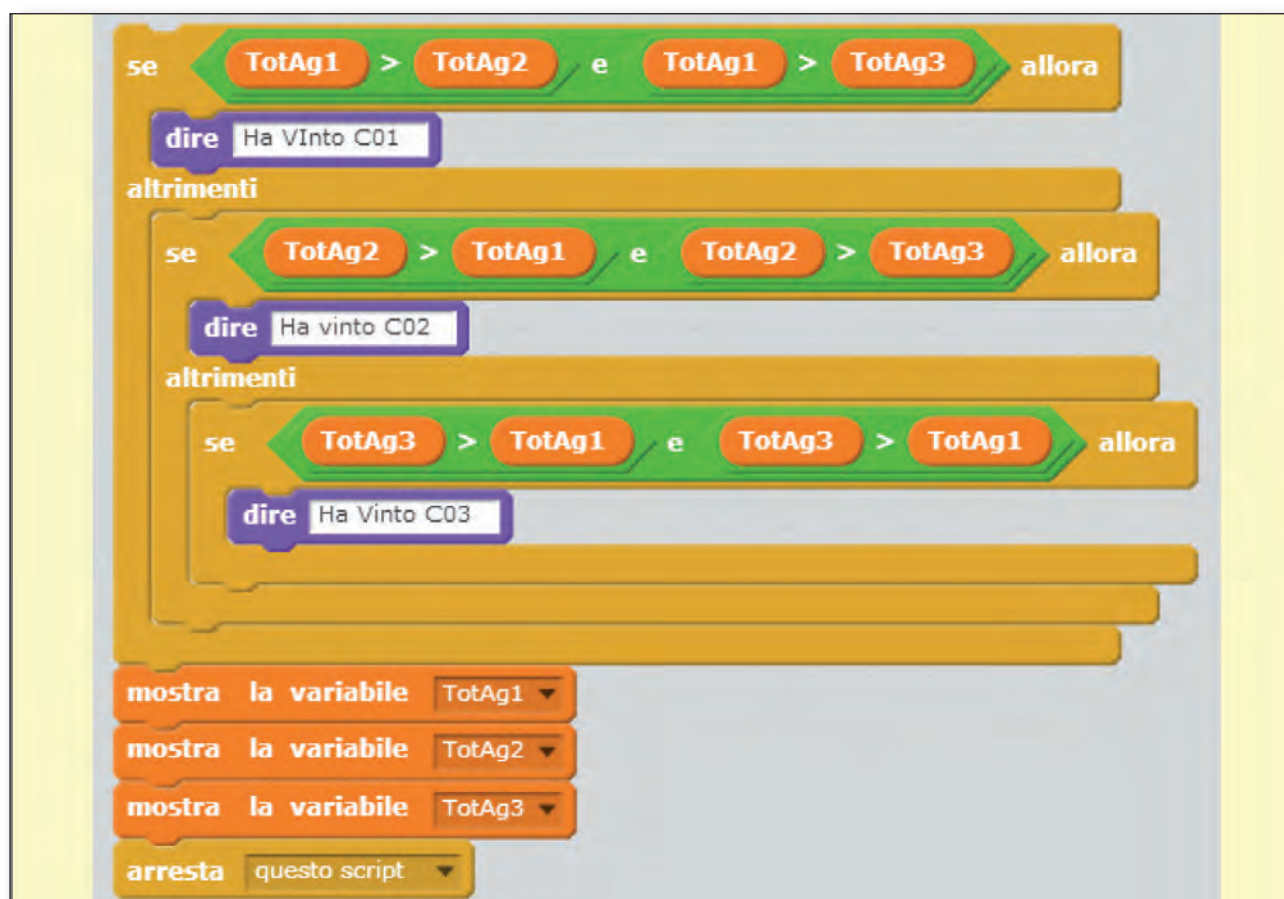
```
Inizio
   'Inizializzazioni
      k=1
      totag1=0
      totag2=0
      totag3=0
      ripeti fino a che k>10         'ciclo
               leggi agente          'leggere una riga tabella
               leggi premio
               se agente="C01" allora „aggiorna totale agente C01
                       totag1=totag1+ premio
               altrimenti
                       se agente="C02" allora „aggiorna totale agente C02
                               totag2=totag2+ premio
                       altrimenti
                               se agente="C03" allora „aggiorna totale agente C03
                                       totag3=totag3+ premio
                               finese
                       finese
               finese
               k=k+1
        fineripeti
   'individuare vincitore
        se totag1>totag2 and totag1>totag3 allora
               scrivi vincitore "C01"
        altrimenti
               se totag2>totag1 and totag2>totag3 allora
                       scrivi vincitore "C02"
               altrimenti
                       se totag3>totag1 and totag3>totag2 allora
                               scrivi vincitore "C03"
                       finese
               finese
        finese
   scrivi totag1, totag2, totag3
fine
```

Codifica Scratch

La codifica viene presentata in tre blocchi contigui, per motivi di spazio e logici.

```
quando si clicca su [bandiera verde]
porta TotAg1 a 0                          Inizializzazioni
nascondi la variabile TotAg1
porta TotAg2 a 0
nascondi la variabile TotAg2
porta TotAg3 a 0
nascondi la variabile TotAg3
porta Agente a
porta Premio a 0
porta k a 1
```

```
ripeti fino a quando k > 5
    chiedi Inserire Agente e attendi
    porta Agente a risposta
    chiedi Inserire premio assicurativo e attendi
    porta Premio a risposta
    se Agente = C01 allora
        porta TotAg1 a TotAg1 + Premio
    altrimenti
        se Agente = C02 allora
            porta TotAg2 a TotAg2 + Premio
        altrimenti
            se Agente = C03 allora
                porta TotAg3 a TotAg3 + Premio
    cambia k di 1
```

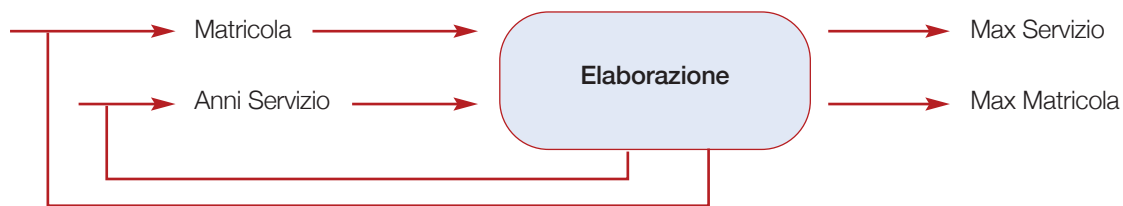
```
se TotAg1 > TotAg2 e TotAg1 > TotAg3 allora
    dire Ha VInto C01
altrimenti
    se TotAg2 > TotAg1 e TotAg2 > TotAg3 allora
        dire Ha vinto C02
    altrimenti
        se TotAg3 > TotAg1 e TotAg3 > TotAg1 allora
            dire Ha Vinto C03
mostra la variabile TotAg1
mostra la variabile TotAg2
mostra la variabile TotAg3
arresta questo script
```

Nell'area stage sarà visibile la seguente situazione:

• *Esempio 4*

In una tabella di cinque righe, sono memorizzati la matricola e gli anni di servizio di altrettanti dipendenti di un'azienda. Individuare il dipendente con più anni di servizio.

Analisi del problema

Il problema proposto implica la ricerca di un valore massimo all'interno di una lista.
Tale valore viene individuato seguendo le seguenti fasi:
inizializzazione del massimo riferito agli anni di servizio confronto di ogni elemento della lista con il massimo, se gli anni di servizio superano il massimo, allora occorre aggiornarlo.
Come al solito analizziamo la condizione di uscita del ciclo essa è: k>5, il corpo del ciclo si ripete finché non diventa vera la frase k>5 , cioè quando k assume il valore 6.
Il corpo del ciclo contiene l'inizializzazione del massimo, quando k=1, in seguito occorre confrontare il valore letto con il massimo per eventualmente aggiornarlo. Nel corpo deve essere introdotta l'incremento del contatore.

Schema di I/O

In questo caso la rappresentazione del ciclo nello schema di I/O è solo grafica ma necessaria per comprendere l'inserimento dei dati d'ingresso finché K>5 non diventa vera

Analisi dei dati

Identificatore	Formato	I/O/W	Significato
Matricola	Carattere	I	Matricola del dipendente
AnniServizio	Numerico	I	Anni di servizio del dipendente
MaxServizio	Numerico	O	Massimo anni di servizio
MaxMatricola	Carattere	O	Matricola corrispondente al massimo degli anni di servizio

Algoritmo di pseudo codifica

```
Inizio
    k=1
    ripeti finchè k>5
        leggi matricola
        leggi anniservizio
        se k = 1 allora
                maxmatricola=matricola
                maxservizio=anniservizio
        altrimenti
                se anniservizio>maxservizio allora
                        maxmatricola=matricola
                        maxservizio=anniservizio
                finese
        finese
        k=k+1
    fineripeti
      scrivi matricola, maxservizio
fine
```

Codifica Scratch

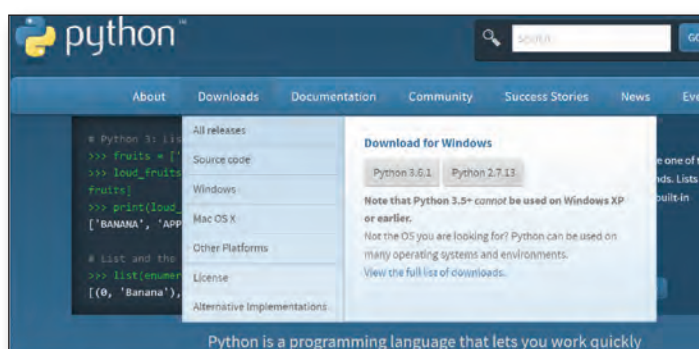
```
quando si clicca su [bandiera verde]
porta Matricola a [ ]                      Inizializzazioni
porta AnniServizio a 0
porta k a 1
ripeti fino a quando k > 5
    chiedi Inserire Matricola e attendi    input matricola
    porta Matricola a risposta
    chiedi Inserire Anni di Servizio e attendi    input anni di servizio
    porta AnniServizio a risposta
    se k = 1 allora                        Inizializzare massimo
        porta MaxMatricola a Matricola
        porta MaxServizio a AnniServizio
    altrimenti
        se AnniServizio > MaxServizio allora    aggiornare massimo
            porta MaxMatricola a Matricola
            porta MaxServizio a AnniServizio
    cambia k di 1                          incrementare k
pensa Hmm... per 3 secondi
dire unione di Matricola del valore massimo: e MaxMatricola per 4 secondi
dire unione di VALORE MASSIMO ANNI SERVIZIO: e MaxServizio
arresta questo script
```

Il linguaggio Python

Autore: **Enrico Sartirana**

Python è un linguaggio di programmazione, ideato da Guido van Rossum negli anni novanta: il suo nome ricorda quello di un pericoloso serpente, in realtà Guido lo scelse perché era appassionato dei Monty Python, degli attori comici inglesi con un particolare humour, che, come riporta il sito ufficiale del linguaggio di programmazione, siete invitati ad andare a cercare su youtube.
A parte il nome, Python si è dimostrato essere un linguaggio con due caratteristiche eccezionali: è semplice ma al contempo molto potente. Uno dei principi del pensiero computazionale è di mantenere basso il livello di complessità di un problema: con Python, grazie alla semplicità del linguaggio, possiamo affrontare problemi complessi.
Non sembrerà quindi strano scoprire che, ad esempio, Python viene studiato nelle università di tutto il mondo per affrontare problemi di Intelligenza Artificiale, o di Robotica. Python trova spazio nelle più diverse applicazioni ed elencarle di seguito sarebbe comunque riduttivo: prova tu a cercare su web in quali ambiti viene utilizzato e dove si pensa che verrà utilizzato.

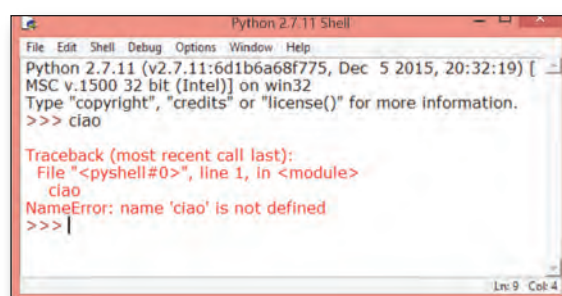
Dove trovo Python

Python è presente in due versioni: la 2.x e la 3.x (dove x può assumere valori che, nel tempo, crescono, ad esempio 2.7). La presenza di due versioni è dovuta al fatto che la 3.x non è "retro-compatibile", non è possibile cioè prendere un programma scritto in Python 2.x ed eseguirlo in ambiente 3.x senza correre il rischio di errori di esecuzione.
La versione più usata è attualmente la 2.x e noi ci soffermeremo su questa; in realtà sono poche le differenze che interessano quello che vedremo.
Per scrivere un programma Python è necessario scaricare l'ambiente Python sul nostro computer: lo possiamo trovare gratuitamente sul sito ufficiale di Python www.python.org: presta solo attenzione a scaricare la versione 2.x, sotto la voce Downloads. Se segui l'installazione consigliata ti ritroverai la cartella, ad es. Python27, nel tuo disco C.

Come usare Python

Se con Scratch eravamo in presenza di un ambiente grafico, con Python siamo in un linguaggio di programmazione tradizionale, dove dobbiamo scrivere istruzioni e vederne l'esecuzione.
Abbiamo due modalità di lavoro: a shell, o a interprete dei comandi, dove scriveremo un'istruzione e la manderemo subito in esecuzione; la seconda modalità è quella classica, dove il nostro programma è una sequenza di istruzioni, scritte all'interno di un file di testo, che potremo successivamente mandare in esecuzione tutte le volte che vorremo.
Esistono diversi strumenti per scrivere un programma, noi utilizzeremo quello offerto insieme al linguaggio Python: IDLE
Scopriremo gli elementi base del linguaggio usando la shell, per poi scrivere i nostri primi programmi.

Esploriamo Python

Dalla voce "Cerca" del tuo Sistema Operativo, digita IDLE (dopo aver installato Python): comparirà la voce IDLE (Python GUI), puoi eseguirla.
Curiosità: GUI è l'acronimo di Graphic User Interface, l'interfaccia grafica a disposizione dell'utente per il tuo programma.
Eseguendo IDLE ci troviamo nella shell di Python, nella prima riga trovi scritta la versione di Python installata, nella riga successiva sono visualizzati tre segni maggiore >>> in attesa del tuo comando: da questo momento possiamo inserire i comandi Python.
Attenzione: un linguaggio di programmazione è composto da poche parole, dovrai imparare queste parole e seguire le rigide regole del linguaggio, senza questa disciplina ti confronterai spesso con risultati come il seguente,

se scriviamo *ciao* sulla shell:
>>> ciao
Traceback (most recent call last):
  File "<pyshell#0>", line 1, in <module>
    ciao
NameError: name 'ciao' is not defined
Detto tra noi significa che Python non sa cosa sia ciao e non sa cosa farsene.

Valori, tipi, variabili, operazioni

Numeri e operazioni

Abbiamo numeri interi e numeri reali: sulla shell puoi scrivere 2 e premere invio, Python ti stampa il numero appena inserito: non ti parrà gran cosa, ma rispetto a prima hai scritto qualcosa che l'interprete Python ha capito, un numero intero.
Puoi anche scrivere un numero lungo a piacere *374572463298472394793847*, Python risponderà con *374572463298472394793847L* dove la L finale indica che il numero è Lungo.
Per scrivere numeri con la virgola, reali,devi utilizzare come separatore decimale il punto: quindi non scriverai 3,14 ma 3.14; fai molta attenzione a questo aspetto, se scrivi *3,14* Python non dà errore ma pensa che tu voglia dire l'elenco dei due numeri *3* e *14*.

Python ci mette a disposizione la funzione type(), attraverso la quale chiedere di che tipo è un valore: prova a scrivere type(2) ti risponderà:
<type 'int'>
che significa che il tipo è un int, un intero.
Se invece scriviamo *type(3.14)* otteniamo:
<type 'float'>
che significa che 3.14 è di tipo float, abbreviazione di floating point, per indicare un numero reale.
Per i numeri negativi basta inserire un – (segno meno) prima del numero.
A questo punto sei già in grado di utilizzare Python come una potente calcolatrice, sapendo che gli operatori aritmetici sono:

- \+ (per la somma) - (per la sottrazione)
- * (per la moltiplicazione)
- / (per la divisione)

Prova alcune operazioni, puoi utilizzare anche le parentesi: usa le parentesi tonde quanto vuoi, basta solo che il numero di parentesi aperte sia uguale a quelle chiuse, e che l'operazione abbia senso.

Attenzione !!!

Abbiamo visto che ci sono numeri interi *(int)* e numeri reali *(float)*: quando Python esegue un'operazione, il risultato di quell'operazione deve appartenere allo stesso dominio dei numeri usati nell'operazione. Per gli operatori di somma, differenza e prodotto non abbiamo problemi, ma l'operatore / di divisone può darci risultati inattesi:
quanto fa 4 / 3 ?
Il risultato sarà 1 , se dividiamo due numeri interi otterremo un risultato intero.
Per conferma puoi anche chiedere a *Python type(4 / 3)* ti dirà che è *int*.
Come faccio ad avere il risultato reale? Basta dire che uno dei due operandi (uno dei due numeri) sia reale: in questo modo forziamo Python a ragionare nel mondo dei numeri reali
4.0 / 3
1.3333333333333333
invece di 4 abbiamo scritto *4.0* dicendo a Python che vogliamo lavorare con i float, e correttamente ci darà il risultato voluto

Ancora due operatori: la divisione intera e il resto

E se vogliamo il risultato della divisione intera tra numeri reali?
Abbiamo il nuovo operatore // che significa divisione intera
4.0 // 3
1.0
Il risultato sarà ancora reale, ma la divisione eseguita è intera, con valore decimale nullo.
Calcolo del resto: l'operatore %
44 gatti in fila per 6 con resto di 2

Abbiamo visto che la divisione tra numeri interi è intera: come abbiamo imparato da piccoli è importante conoscere il resto, Python ce lo offre con l'operatore %

44 / 6
7
44 % 6
2
*6 * 7 + 2*
44

la prima operazione divide 44 per 6 ed ottiene 7
la seconda operazione chiede il resto della divisione intera di 44 / 6, ed ottiene 2
l'ultima operazione esegue sei per sette quarantadue, più due quarantaquattro.

Un altro tipo di dato importante: le stringhe

Abbiamo trattato con i numeri, ma certamente un'altra tipologia di informazione che ci è utile gestire è quella testuale: la possibilità cioè di gestire un nome, un indirizzo, la materia che stiamo seguendo. In Python, come in molti altri linguaggi di programmazione, l'informazione testuale viene indicata di tipo string: una string è una qualsiasi sequenza di caratteri, delimitata tra apici o doppi apici (l'importante è che siano sempre gli stessi).

'Elena'
"Emiliano"

sono tutte stringhe corrette: prova ad aprire IDLE e digitare (sempre tra apici)
type("Elena")
ritornerà <*type* '*str*'> dove *str* indica il tipo string

Operatori sulle stringhe: concatenazione e ripetizione

E' possibile avere anche degli operatori sulle stringhe: in particolare possiamo vedere l'operatore di concatenazione + (il simbolo più)
"Ciao " + "Giuseppe"
ci darà "*Ciao Giuseppe*"

Abbiamo inoltre l'operatore di ripetizione * (asterisco)
*"strano" * 4*
ci darà "*stranostranostranostrano*"

```
Python 2.7.11 Shell
File Edit Shell Debug Options Window Help
>>> 12 / 5
2
>>> 12.0 / 5
2.4
>>> 12.0 // 5
2.0
>>> 12 % 5
2
>>> 12 % 8
4
>>> 'Elena'
'Elena'
>>> "Ciao " + "Elena"
'Ciao Elena'
>>> "Ciao Elena " * 4
'Ciao Elena Ciao Elena Ciao Elena Ciao Elena '
>>>
Ln: 21 Col: 4
```

Attenzione !!!

Quello che scrivi nella shell viene colorato in modo diverso a seconda di cosa scrivi: così un numero avrà un colore diverso di una stringa, o di una funzione: non è importante ricordare ogni colore, l'importante è notare che se sbagliamo a scrivere qualcosa non assumerà il colore adeguato. I colori aiutano quindi a scoprire eventuali errori di scrittura.

Le variabili

Un ultimo ingrediente ed abbiamo tutto per le nostre ricette. Abbiamo visto come rappresentare dei valori, siano essi numeri interi o reali, o stringhe; ci serve un luogo dove memorizzare i nostri valori per poterli utilizzare: tecnicamente sono delle aree di memoria dove scrivere dentro i valori, in pratica possiamo pensarli come dei post-it, dei bigliettini dove scriverci dentro, o dei cassettini in cui depositare i valori.

Ogni variabile la identifichiamo con un nome, che decidiamo noi: per dare il nome alla variabile dobbiamo seguire queste regole:

- *deve iniziare con una lettera o un underscore '_'*
  nome, _indirizzo sono nomi corretti
  2nome, £soldi non sono corretti
- segue una sequenza di lettere, numeri o '_'
  nome_2gatto, codice_fiscale sono corretti
  razze cani, prezzo-totale non sono corretti
- i nomi delle variabili sono case sensitive, cioè sensibili a maiuscole e minuscole
  prova e *Prova* sono due nomi di variabili diverse.

```
>>> nome_figlia = "Elena"
>>> nome_figlio = "Emiliano"
>>> nome padre = "Enrico"
SyntaxError: invalid syntax
>>>
Ln: 34 Col: 4
```

Le prime due variabili hanno un nome corretto, la terza no: perché ?

Una delle caratteristiche rilevanti di un programma scritto bene è la leggibilità del codice, fare in modo cioè che un altro programmatore, o noi stessi a distanza di tempo, siamo in grado di capire cosa faccia il nostro programma: la prima regola per la leggibilità è scegliere nomi di variabili evocative, che abbiano cioè significato.
Saranno preferibili quindi variabili con nomi tipo:
sconto, nome_cliente, indirizzo, interesse_lordo, imposta_dovuta, sommatoria, …
rispetto nomi di variabili del tipo:
a, b, c, z, k02, NA1, … (nomi da evitare perché non evocativi)

Come facciamo a scrivere un valore in una variabile ? L'istruzione di assegnamento
In Python una variabile viene creata nel momento in cui le si assegna un valore:
primo_addendo = 22
questa istruzione crea la variabile di nome *primo_addendo*, a cui viene assegnato il valore 22.

Attenzione !!!

L'operatore = non ha lo stesso significato della matematica, di eguaglianza, ma di assegnamento: viene calcolata l'espressione alla sua destra ed il risultato inserito nella variabile alla sua sinistra.
Ad esempio:

```
>>> valore = 5
>>> valore = valore + 1
>>> valore
6
>>>
```

valore = 5
valore = valore + 1
nella prima istruzione viene creata la variabile *valore* a cui viene assegnato il valore *5*.
nell'istruzione successiva al numero contenuto nella variabile *valore (5)* viene sommato *1*: il risultato, *6*, viene quindi assegnato alla variabile *valore*.
Al termine dell'esecuzione delle due istruzioni la variabile valore contiene il valore 6.

Attenzione !!!

Una variabile, in un determinato istante, può contenere un solo valore: non è possibile risalire a valori precedenti di una variabile.

Il tipo della variabile è determinato dal tipo del valore contenuto:

```
>>> valore = 'caio'
>>> type(valore)
<type 'str'>
>>> valore = 5
>>> type(valore)
<type 'int'>
>>>
```

valore = 'caio'
type(valore) restituirà 'str'
valore = 5
type(valore) restituirà 'int'
durante l'esecuzione del programma una variabile potrà quindi cambiare tipo.

Commenti nel codice
In ultimo, parlando di leggibilità del codice, impareremo ad inserire commenti nei nostri programmi, attraverso l'uso di #
 # questo è un commento
somma = 3 + 4 # sommo i due valori
tutto quello che scriviamo dopo #, fino a fine riga, è ignorato da Python ed ha lo scopo di commento per il programmatore.

```
>>> lato_maggiore = 7   # misura del lato maggiore
>>> lato_minore = 4     # misura del lato minore
>>> area_rettangolo = lato_maggiore * lato_minore
>>> # calcolo dell'area del rettangolo
>>> area_rettangolo
28
>>>
```

Il nostro primo programma
Possiamo scrivere il nostro primo programma Python, come sequenza di tre istruzioni nell'ambiente shell:
nome = 'Enrico'
saluti = 'ciao ' + nome
saluti
'ciao Enrico'
la prima istruzione crea la variabile *nome* e le assegna il valore 'Enrico', di tipo *string*
la seconda istruzione crea la variabile *saluti*, a cui assegna il risultato dell'espressione *'ciao ' + nome*
la terza istruzione mostra il contenuto della variabile *saluti*.

Input, Elaborazione, Output

Come richiedere l'inserimento di un valore all'utente? Come stamparlo?
Per quanto riguarda l'input, a seconda del tipo di dato da inserire abbiamo due istruzioni:

1. Acquisizione di un valore numerico da tastiera
temperatura = input("Inserisci il valore di temperatura rilevato: ")

Inserisci il valore di temperatura rilevato: 23.5
temperatura
23.5

```
>>> temperatura = input("Inserisci il valore della temperatura rilevato: ")
Inserisci il valore della temperatura rilevato: 23.5
>>> temperatura
23.5
>>>
```

La funzione *input* permette di acquisire un valore numerico *(int o float)*, attraverso la stampa a video del messaggio scritto tra parentesi. Il valore inserito da tastiera viene memorizzato nella variabile *temperatura*.

2. Acquisizione di un valore di tipo string
nominativo = raw_input("Come ti chiami ? ")
Come ti chiami ? Luigi
nominativo
'Luigi'
La funzione *raw_input()* permette di acquisire un valore di tipo *string*.
Nell'ultimo programma scriveremo quindi la prima riga; una volta premuto invio, la shell di Python ci stamperà la richiesta inserita tra parentesi, *Come ti chiami ?*, attendendo l'inserimento da tastiera; una volta inserito il nome voluto lo ritroveremo memorizzato nella variabile *nominativo*.

Attenzione !!!

Se acquisisco un numero attraverso la funzione *raw_input()* verrà inserito come *string*, tra virgolette: il valore '*2*' è diverso dal valore *2*, il primo è il carattere '*2*' , il secondo è il numero *2*.
Per comprendere meglio, prova ad eseguire le due seguenti operazioni:

'*2*' + '*3*'
2 + 3
la prima ti restituirà '*23*', la seconda *5*: hai capito perché ?
Nel primo caso ho la stringa '*2*' che viene concatenata alla stringa '*3*', ed il risultato è '*23*'
Nel secondo caso ho il numero *2* a cui viene sommato il numero *3*, ed il risultato è *5*.

Conversione di tipi
Come hai visto '*2*' e *2* per Python sono due cose diverse, e se provi a sommarli ti ritrovi un errore perché Python non sa cosa tu voglia fare con una stringa ed un intero.
Per risolvere l'ambiguità dobbiamo dire a Python cosa vogliamo fare: possiamo chiedere di trasformare un valore di un tipo in un altro tipo, se questo ha significato.
int('3')   # trasforma la stringa '3' nell'intero 3
float('3')   # trasforma la string '3' nel numero reale 3.0
str(2)   # trasforma il valore intero 2 nella stringa '2'

Grazie alle funzioni di conversione di tipo possiamo quindi scrivere i seguenti esempi:
'2' + str(2)   # otteniamo la stringa '22'
int('2') + 2   # otteniamo il valore intero 4
numero = 3
frase = 'Ci sono ' + str(numero) + ' fiori nel vaso'
nell'ultima riga, essendo la variabile *numero* di tipo intero, non avremmo potuto scrivere la frase senza utilizzare la funzione di conversione *str()*.

Output: il comando print

Nella shell, per conoscere il valore di una variabile basta digitarne il nome

x = 7

x

7

purtroppo quando si scrive un vero programma in un file non è possibile visualizzare in questo modo il contenuto di una variabile; a questo scopo è necessario utilizzare il comando print seguito da quello che si vuole stampare:

```
>>> print 'ciao'
ciao
>>> print 7 *3
21
>>> print 'ciao'
ciao
>>> print 7 * 3
21
>>> print " 7 * 3 = " + str(7 * 3)
 7 * 3 = 21
>>>
```

print 'ciao'   # stampa ciao

*print 7 * 3   # stampa 21*

*print " 7 * 3 = " + str(7 * 3)   # stampa 7 * 3 = 21*

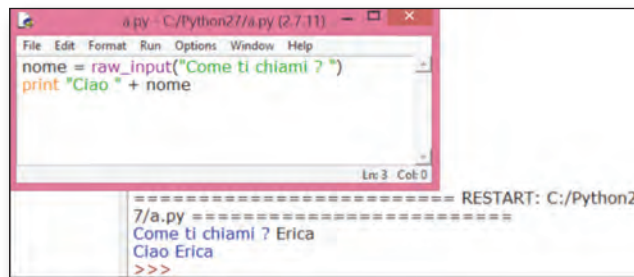
Il comando *print* è in grado di interpretare correttamente stringhe e numeri, facendo anche eseguire calcoli al suo interno; se invece, come nel terzo esempio, abbiamo concatenazione di stringhe e valori numerici, occorre utilizzare la funzione di conversione *str()*.

Scrivere un programma

Fino a questo punto abbiamo imparato Python attraverso l'utilizzo della shell, proviamo adesso a scrivere un vero programma: per fare questo dobbiamo, dalla shell di IDLE, selezionare la voce di menu File, New File

Qui possiamo scrivere le istruzioni per il nostro programma:

nome = raw_input("Come ti chiami ? ")

print "Ciao " + nome

Per eseguirlo basterà selezionare, dalla voce di menù Run, Run Module o premere il tasto di scelta rapida F5. Come prima cosa ti viene richiesto di salvare il tuo programma: in questo modo potrai recuperare ed eseguire il tuo programma tutte le volte che vorrai; il file viene salvato con estensione .py, che indica un programma Python.

Ora puoi vedere in esecuzione il tuo programma:

======================== RESTART: C:\Python27\a.py ========================

Come ti chiami ? Erica

Ciao Erica

Programmi in Python

Possiamo ora seguire gli stessi programmi visti in Scratch, affrontati questa volta in Python.

<u>Esempio 1:</u> Scrivi un programma per calcolare l'area di un rettangolo conoscendo le misure della base e dell'altezza: dobbiamo quindi acquisire i due valori di base e altezza tramite la funzione input, calcolare l'area in base alla formula area = base x altezza e stampare il risultato.

 # Area rettangolo: calcolo dell'area a partire dalle misure di base e altezza

 # Prima versione: 20/04/2017

 # Autore: Enrico

base = input("Inserisci la misura della base del rettangolo: ")

altezza = input("Inserisci adesso la misura dell'altezza del rettangolo: ")

*area_rettangolo = base * altezza   # calcolo area secondo la formula*

print "L'area del rettangolo di base " + str(base) + " e altezza " + str(altezza) + " vale " + str(area_rettangolo)

Come dall'esempio riportato, è buona cosa far precedere al codice vero e proprio delle righe di commento, indicando cosa fa il programma, quando è stato scritto e chi lo ha scritto.

Attenzione !!!

Il programma appena scritto funziona correttamente con valori numerici interi o reali; va in errore nel caso l'utente inserisca una lettera invece di un numero.

Esempio 2: Dato il lato di un quadrato, scrivere un programma che stampi area e perimetro
Per questo esercizio dobbiamo ricordare come richiedere un valore, oltre alle due formule della geometria per il calcolo dell'area e del perimetro di un quadrato.

```
# Calcolo di area e perimetro di un quadrato
# Prima versione: 20 Aprile 2017
# Autore: Enrico
lato_quadrato = input("Inserisci la misura del lato di un quadrato: ")
area_quadrato = lato_quadrato * lato_quadrato
perimetro_quadrato = lato_quadrato * 4
print "Vuoi sapere i dati di un quadrato di lato " + str(lato_quadrato)
print " *** Area = " + str(area_quadrato)
print " *** Perimetro = " + str(perimetro_quadrato)
```

Puoi notare la semplicità del programma, che non richiede pseudo-codifica in quanto Python è talmente semplice da rappresentare già adeguatamente la soluzione del problema in termini sintetici.

Esempio 3: Dati tre numeri, progettare un programma in grado di calcolarne la media aritmetica
Per questo problema, che richiede l'inserimento di tre valori ed il calcolo della media, quale somme dei tre valori e successiva divisione per tre, dobbiamo ricordare il comportamento di Python rispetto la divisione: per evitare la divisione intera dovremo dire a Python di ragionare in termini reali (float), per fare questo basterà richiedere la divisione per 3.0 anziché 3. Prova a risolverlo tu.

Operare una scelta: la selezione

In Python, per determinare una scelta tra due possibilità, viene utilizzato il costrutto *if* secondo i seguenti schemi:

schema1:

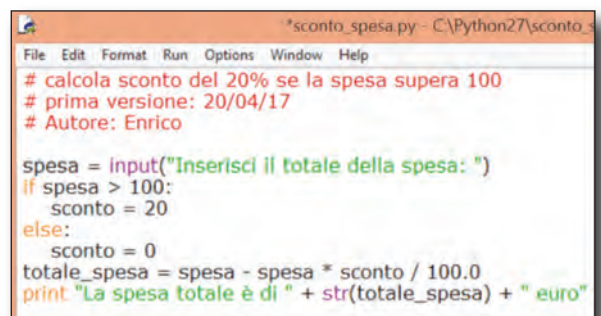
if condizione:
        blocco condizionale
resto del codice

schema2:
if condizione:
        blocco condizionale
else:
        blocco alternativo
resto del codice

schema3:
if condizione1:
        blocco condizione1
elif condizione2:
        blocco condizione2
...
elif condizioneN:
        blocco condizioneN
else:
        blocco alternativo
resto del codice

Gli aspetti su cui prestare attenzione sono la condizione e la sintassi corretta per il codice; attenzione ai due punti (:) dopo la condizione e al rientro del codice (un tab) nelle righe successive; ogni blocco può essere formato da una o più righe.

Esempio 3: In un negozio se si acquista merce per più di 100 euro si ha diritto ad uno sconto del 20%. Progettare un'applicazione in grado di visualizzare la spesa scontata.

```
*sconto_spesa.py - C:\Python27\sconto_s
File  Edit  Format  Run  Options  Window  Help
# calcola sconto del 20% se la spesa supera 100
# prima versione: 20/04/17
# Autore: Enrico

spesa = input("Inserisci il totale della spesa: ")
if spesa > 100:
    sconto = 20
else:
    sconto = 0
totale_spesa = spesa - spesa * sconto / 100.0
print "La spesa totale è di " + str(totale_spesa) + " euro"
```

La condizione

La condizione è un'espressione che può essere vera o falsa: quando la condizione è vera si dice che è soddisfatta e viene eseguito il blocco di codice che segue l'*if*.

Una condizione è un'espressione di confronto che può utilizzare i seguenti operatori:

>	maggiore
<	minore
>=	maggiore o uguale (va scritto senza spazi e con prima il >)
<=	minore o uguale (va scritto senza spazi e con prima il <)
==	uguale (senza spazi, attenzione = significa assegna, == è l'operatore di confronto)
!=	diverso da

Esempio 4: chiedi un numero e stampa se è pari o dispari.

Attenzione !!!

Sappiamo che un numero è pari se è divisibile per 2: quindi se lo dividiamo per 2 il resto sarà nullo. Possiamo quindi utilizzare l'operatore %, che ci dirà qual è il resto della divisione per 2.

```
valore = input("Inserisci il valore: ")
if valore % 2 == 0:
        print str(valore) + " è un numero pari"
else:
        print str(valore) + " è un numero dispari"
```

<u>Esempio 5:</u> equazione di primo grado

 # equazione di primo grado
 # ax = b
a = input("Inserisci il valore del parametro a: ")
b = input("Inserisci il valore del parametro b: ")
if a == 0:   # se a è uguale a zero
    if b == 0:   # se anche b è uguale a zero
        print "equazione indeterminata"
    else: # caso in cui a è uguale a zero e b è diverso da zero
        print "equazione impossibile"
else: # a e b diversi da zero
    *print 1.0 * b / a   # attenzione: 1.0 per forzare l'operazione come float*

Che tipo è il risultato di un'espressione condizionale ?

Se sulla shell proviamo a scrivere

2 > 3
False
Ci dice False
Se poi scriviamo
type(2 > 3)
<type 'bool'>

Cosa indicano queste informazioni ? Nello studio dell'informatica è sempre importante fare delle ipotesi e confrontarle con quanto ci dice lo strumento automatico:
in questo caso siamo in presenza di un nuovo tipo di dati 'bool', oltre a quelli che conosciamo già (int, float, string).

Quali sono i valori possibili del tipo bool ? Proviamo: abbiamo solo due possibilità:
2 > 3 ci restituisce *False*
2 < 3 ci restituisce *True*
Quando scriviamo una condizione abbiamo solo due possibilità: che sia vera *(True)* o falsa *(False)*, e questi sono i due possibili valori dell'algebra booleana: George Boole fu un matematico britannico, che studiò la logica e le regole per lo studio di verità delle proposizioni. Nel diciannovesimo secolo, quando le studiò, erano teorie matematiche; quando, cent'anni dopo, iniziò l'era dell'informatica, le sue teorie servirono ad affrontare molti dei problemi posti dalla nuova scienza.
Gli operatori di confronto servono per scrivere espressioni condizionali, esistono invece gli operatori logici che servono per comporre più espressioni condizionali: gli operatori logici di base sono l'operatore unario NOT e gli operatori binari AND e OR, che seguono le regole delle tavole di verità, già viste per Scratch, che noi possiamo direttamente chiedere a Python:

```
>>> not True
False
>>> not False
True
>>> True and True
True
>>> True and False
False
>>> False and True
False
>>> False and False
False
>>> True or True
True
>>> True or False
True
>>> False or True
True
>>> False or False
False
>>>
```

not True   # restituisce False
not False   # restituisce True
True and True   # restituisce True
True and False   # restituisce False
False and True   # restituisce False
False and False   # restituisce False
True or True   # restituisce True
True or False   # restituisce True
False or True   # restituisce True
False and False   # restituisce False

Possiamo comprendere il significato di queste espressioni con degli esempi:
L'operatore not è la negazione: se, ad esempio, sta piovendo
not piove mi darà False perché dice che non sta piovendo.

Se piove, ma non c'è vento avrò che:
piove and vento sarà *False*, perché per essere vera dovrebbe piovere ed esserci il vento;
piove or vento sarà *True*, perché sto dicendo che piove o c'è il vento, e quindi basta che una delle due affermazioni sia vera.

```
>>> piove = True
>>> vento = False
>>> piove and vento
False
>>> piove or vento
True
>>>
```

Ripetere più volte le istruzioni: i cicli, o loop

La ripetizione di un blocco di codice, o iterazione, è la terza ed ultima struttura di controllo di un linguaggio di programmazione, dopo la sequenza (un'istruzione dopo l'altra) e la selezione (che abbiamo appena visto).
In Python abbiamo due possibilità per esprimere un ciclo: vediamo il primo.

```
for i in range(5):
	print "Ciao"
```

stamperà cinque volte "*Ciao*", ripetendo quindi per 5 volte il blocco di codice che segue l'istruzione *for*; come per l'istruzione *if*, per indicare qual è il blocco che dipende dalla condizione, occorre indentare le istruzioni successive, cioè farle rientrare con un tab.

Per comprendere meglio quanto sia importante l'indentazione osserva i due esempi:

```
valore = 0
for numero in range(5):
	valore = valore + 1
	print valore
```

In questo caso vengono stampati tutti i valori da 1 a 5, perché il comando print viene invocato all'interno del blocco del *for*: verrà quindi eseguito ad ogni giro.

```
valore = 0
for numero in range(5):
	valore = valore + 1
print valore
```

In questo caso, differente dal precedente, verrà stampato solo il valore 5, perché il comando print verrà eseguito solo dopo aver terminato l'intero ciclo *for*.

Attenzione !!!

Il costrutto *for i in range(5)* ripete per 5 volte il blocco di istruzioni successive; può essere utile sapere che, per contare 5 volte, il nostro sistema non va da 1 a 5, ma da 0 a 4; può sembrare strano, ma praticamente in tutti i linguaggi di programmazione è così, per motivi tecnici che puoi approfondire sul web.
Come al solito, non è necessario crederci ciecamente, basta chiederlo a Python in questo modo:

```
for i in range(5):
	print i,
```

che stamperà 0 1 2 3 4 (la virgola alla fine del print indica di non andare a capo ad ogni stampa)
Descriviamo l'istruzione for appena scritta:

- *for* è la parola riservata che indica l'inizio di un'istruzione di ciclo
- *i* è il nome della variabile che ci servirà per contare, può avere qualsiasi nome di variabile consentito (ad esempio *numero*, o *valore*)
- *in* è un'altra parola riservata di Python, che indica che il valore della variabile va preso nell'insieme di valori successivo
- *range(5)* è una funzione che genera la lista di valori da 0 a 4; puoi provare anche nella shell a scrivere *range(5)*, otterrai *[0, 1, 2, 3, 4]*

Riassumendo, abbiamo la variabile *i* che assume, uno alla volta, tutti i valori compresi nell'insieme [0, 1, 2, 3, 4] e, per ogni valore che assume, esegue il blocco di istruzioni sottostante il ciclo *for*.

Ciclo for per scandire una parola

Una parola può essere trattata per Python come l'elenco delle lettere che la compongono: così, possiamo scrivere il seguente codice:

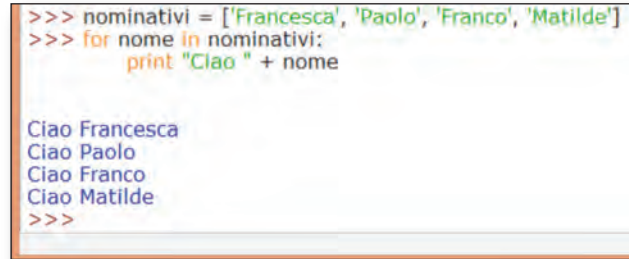
```
for lettera in "albero":
	print lettera,
```

Otterremo *a l b e r o*, che non è semplicemente la parola con spazi in mezzo, ma è l'estrazione di ogni lettera che compone la parola albero.

Ciclo for in elenchi qualsiasi

Abbiamo visto che *range(5)* crea l'elenco *[0, 1, 2, 3, 4]* che in Python si chiama lista ed è contenuto tra parentesi quadre. Allo stesso modo possiamo dare in pasto ad un ciclo for una lista qualsiasi.

nominativi = ['Francesca', 'Paolo', 'Franco', 'Matilde']

for nome in nominativi: print "Ciao " + nome

per ottenere

Ciao Francesca
Ciao Paolo
Ciao Franco
Ciao Matilde

```
>>> nominativi = ['Francesca', 'Paolo', 'Franco', 'Matilde']
>>> for nome in nominativi:
        print "Ciao " + nome


Ciao Francesca
Ciao Paolo
Ciao Franco
Ciao Matilde
>>>
```

Un altro tipo ciclo: il ciclo condizionato, o ciclo while

Con il ciclo *for* possiamo eseguire un blocco di istruzioni un numero definito di volte, o scandire una parola, o una frase, o una lista di elementi.
A volte è invece necessario ripetere un blocco di istruzioni fino a quando non accade qualcosa: ad esempio inserisci le monete fino a raggiungere la cifra desiderata, oppure ripeti quelle istruzioni fino a quando non decidi di terminare, o lancia i dadi finché non termina la partita.
Tutti questi casi vengono indicati come cicli condizionati: come per la selezione dovremo quindi impostare una condizione, il soddisfacimento della quale provocherà l'esecuzione del blocco di istruzioni contenuto nel ciclo. A differenza della selezione però, dove il blocco di istruzioni sottostante viene eseguito al più una volta sola, per quanto riguarda il ciclo condizionato il blocco di istruzioni verrà eseguito finché la condizione non risulterà falsa.

```
scelta = 1
while scelta != 0:
        scelta = input("Digita un numero per continuare, 0 per uscire: ")
print "Grazie, ci vediamo la prossima volta"
```

esempio di esecuzione

Digita un numero per continuare, 0 per uscire: 2
Digita un numero per continuare, 0 per uscire: 3
Digita un numero per continuare, 0 per uscire: 4
Digita un numero per continuare, 0 per uscire: 1
Digita un numero per continuare, 0 per uscire: 0
Grazie, ci vediamo la prossima volta

Con un ciclo while possiamo realizzare lo stesso comportamento di un ciclo for

```
indice = 0
while indice < 5:
        print indice,
        indice = indice + 1
```

Come puoi vedere, sono richieste più righe di codice rispetto al più semplice ciclo for:
nella prima riga *(indice = 0)* dobbiamo inizializzare a 0 una variabile che chiamiamo indice (sarà analoga alla i del for i in range(5)),
segue quindi la definizione del ciclo condizionato (*while indice < 5:*), dove alla parola riservata *while* segue la condizione *indice < 5*, i due punti successivi indicano che sta iniziando un blocco di codice; finché *indice* conterrà un valore inferiore a 5, verrà eseguito il blocco di codice: dopo ogni ripetizione verrà rivalutata la condizione.
con *print indice*, stampiamo il valore corrente di indice e, mettendo la virgola, non va a capo per la stampa successiva,
col ciclo for avremmo già finito, nel nostro caso dobbiamo invece aggiungere un ultima istruzione *(indice = indice + 1)* per incrementare il valore della variabile indice.

Attenzione !!!

L'ultima istruzione (l'incremento della variabile *indice*) è di vitale importanza e, associata alla condizione, ci protegge da uno degli eventi più perniciosi che possano accadere nell'esecuzione di un programma: il loop infinito. Il loop infinito, o ciclo infinito, accade quando la nostra condizione non sarà mai falsa: nel nostro esempio, se io non incremento il valore contenuto nella variabile *indice*, la condizione *indice* < *5* non diventerà mai *False*. La mia variabile *indice* varrà sempre 0 e continuerò all'infinito a stampare 0.

Esempio: scrivi un programma che chiede all'utente un numero e stampa se è pari, dopo la stampa richiede se inserire un nuovo valore o terminare (i numeri a inizio riga servono per descrivere il codice, non sono presenti nel programma)

```
1.      scelta = 1
2.      while scelta != 0:
3.              numero = input("Inserisci un numero, ti dirò se è pari: ")
4.              if numero % 2 == 0:
5.                      print str(numero) + " è un numero pari"
6.              else:
7.                      print str(numero) + " è un numero dispari"
8.              scelta = input("Se vuoi continuare premi 1 , se vuoi terminare premi 0: ")
```

Esempio di esecuzione:
Inserisci un numero, ti dirò se è pari: 3
3 è un numero dispari
Se vuoi continuare premi 1 , se vuoi terminare premi 0: 1
Inserisci un numero, ti dirò se è pari: 4
4 è un numero pari
Se vuoi continuare premi 1 , se vuoi terminare premi 0: 1
Inserisci un numero, ti dirò se è pari: 5
5 è un numero dispari
Se vuoi continuare premi 1 , se vuoi terminare premi 0: 0

```
>>> scelta = 1
>>> while scelta != 0:
        numero = input("Inserisci un numero, ti dirò se è pari: ")
        if numero % 2 == 0:
                print str(numero) + " è un numero pari"
        else:
                print str(numero) + " è un numero dispari"
        scelta = input("Se vuoi continuare premi 1, se vuoi terminare premi 0: ")


Inserisci un numero, ti dirò se è pari: 3
3 è un numero dispari
Se vuoi continuare premi 1, se vuoi terminare premi 0: 1
Inserisci un numero, ti dirò se è pari: 4
4 è un numero pari
Se vuoi continuare premi 1, se vuoi terminare premi 0: 1
Inserisci un numero, ti dirò se è pari: 5
5 è un numero dispari
Se vuoi continuare premi 1, se vuoi terminare premi 0: 0
>>>
```

riga 1: inizializziamo il valore della variabile *scelta* a 1, ci permetterà di entrare la prima volta nel ciclo,
riga 2: con la parola riservata *while* e la condizione scelta *!= 0* chiediamo di eseguire il blocco seguente finché il valore di *scelta* è diverso da zero; attenzione: guarda subito l'ultima istruzione all'interno del blocco sottostante il ciclo while, alla riga 8, dove chiediamo all'utente di digitare 0 se vuole uscire: questa è la nostra assicurazione contro il loop infinito; è buona regola inserire subito l'istruzione che determinerà l'uscita dal ciclo, poi le altre istruzioni le possiamo inserire in mezzo;
riga 3: con *numero = input("Inserisci un numero, ti dirò se è pari: ")* chiediamo all'utente di inserire il valore da trattare;
riga 4: segue quindi l'istruzione condizionale, che fa eseguire la riga 5 se è soddisfatta la condizione
numero % 2 == 0, cioè se il resto della divisione intera di *numero / 2* vale zero, cioè se numero è pari
riga 5: stampa che numero è pari
riga 6: inizio del blocco else, che viene eseguito se la condizione dell'*if* risulta essere falsa, se cioè il numero non è pari
riga 7: stampa che il numero è dispari
riga 8 chiede all'utente se vuole continuare o terminare: il valore acquisito nella variabile *scelta* viene riutilizzato nella condizione del *while*, alla riga 2, per determinare se il blocco di istruzioni debba essere rieseguito o no. Questa condizione verrà rivalutata finché non risulterà *False*, determinando l'uscita dal ciclo e la terminazione del codice, o l'esecuzione delle istruzioni successive il blocco del *while*.

Un esempio di algoritmo interessante da realizzare in Python è quello del Massimo Comun Divisore risolto con l'algoritmo di Euclide, l'analisi e la realizzazione del programma è mostrata nel video collegato al codice QR o al link equivalente:

Algoritmo di Euclide
MCD con Python
http://qrbridge.me/9n83

Conclusioni

Abbiamo fatto una galoppata nel mondo della programmazione: Python viene utilizzato per imparare a scrivere codice, ma anche per complessi motori di ricerca nel web, o sistemi di apprendimento automatico. Quello che abbiamo appreso in queste pagine sono solo i mattoncini base, ma che ti permettono già di scrivere semplici programmi e vederli funzionare: non è poco, forse è la prima volta che passi dall'altra parte, che non sei solo un utilizzatore di software, ma che lo crei tu, lo progetti, lo scrivi, lo correggi e lo vedi funzionare. Il tuo pensiero che prende forma in un programma e che viene eseguito dal computer. Puoi riprendere gli esercizi proposti per Scratch ed affrontarli in Python, puoi cercare nel web approfondimenti, per esempio sull'utilizzo del modulo turtle per realizzare applicazioni grafiche in Python: come abbiamo detto in principio del capitolo non è solo scrivere codice, è raffinare il tuo pensiero, esercitarlo, acquisire nuovi modelli, adesso hai la chiave.

Esercizi

Sequenza

1. Di un trapezio rettangolo si conoscono le misure delle due basi e dell'altezza. Progettare un'applicazione che determini area e perimetro.
2. Dato il raggio di un cerchio progettare un'applicazione per determinarne l'area e la circonferenza.
3. Di un trapezio isoscele sono determinati le due basi e il lato obliquo. Progettare un'applicazione per calcolare perimetro e area.
4. Di un triangolo si conoscono le misure dei lati. Progettare un'applicazione in grado di determinarne l'area attraverso la formula di Erone: http://it.wikipedia.org/wiki/Triangolo
5. Dati due punti nel piano cartesiano A(XA,YA) e B(XB,YB). Progettare un' applicazione in grado di determinare distanza e coordinate del punto medio M.
6. Dati tre punti A, B, C distinti, nel piano cartesiano, progettare un'applicazione in grado di determinarne il perimetro e il baricentro del triangolo ABC.
7. Dati tre numeri N1, N2, N3, determinare la loro media aritmetica e la percentuale che ciascuno occupa rispetto al totale.
8. Un negozio vende le arance a 2 euro al kg. Il costo sostenuto dal negozio invece è di 75 centesimi al kg. Analogamente per le mele che vengono vendute a 1,5 €/kg mentre costano 50 cent/kg. Al termine della giornata risultano venduti X kg di arance e Y kg di mele. Progettare un'applicazione in grado di determinare il guadagno sulle arance e sulle mele, nonché il guadagno totale.
9. Un operaio specializzato sta eseguendo alcuni lavori di manutenzione per un'azienda, ad una tariffa di 60 € lordi all'ora. Ha cominciato a lavorare ad una certa ora HInizio ed ha terminato alle ore HFine. Scrivere un'applicazione in grado di determinare il numero di ore lavorate. Il totale lordo e quindi il salario netto sapendo che viene applicata una trattenuta del 27%.
10. dato il prezzo unitario di un prodotto, e la quantità acquistata, determinare il totale da pagare, sapendo che su esso occorre applicare una certa aliquota IVA.

Selezione

11. Data l'età di una persona, progettare un'applicazione in grado di determinare l'età e quindi se si tratta o meno di un maggiorenne 12. Di due persone viene fornito il relativo anno di nascita. Determinare le età di ciascuno e quindi visualizzarle in ordine crescente.
13. Dato un numero intero positivo, determinare se è pari o dispari.
14. E' il momento dei saldi e ogni negozio applica i suoi; in particolare ve ne è uno che applica il 30% di sconto sulla spesa totale se questa supera gli 80 euro, altrimenti lo sconto è del 10%. Progettare un'applicazione in grado di determinare e visualizzare il prezzo totale, il valore dello sconto e il prezzo scontato.
15. Due squadre A e B che in classifica hanno rispettivamente PA e PB punti, disputano un incontro di calcio.
    L'incontro termina con GA goal per la squadra A e GB goal per la B.
16. Progettare un'applicazione che acquisito il punteggio finale sia in grado di aggiornare i punti in classifica delle squadre e visualizzarli. Data l'equazione di una retta: $ax + by + c = C$
    e le coordinate di un punto P(Xp, Yp), progettare un'applicazione che verifichi se P appartiene o meno alla retta.
17. Data un'equazione di secondo grado: $a^2x + bx + c = C$
    scrivere un' applicazione in grado di determinarne la soluzione.
18. Dato il sistema composto dalle equazioni:
    $a_1x + b_1y = c_1$
    $a_2x + b_2y = c_2$
    Progettare un'applicazione in grado di risolvere il sistema applicando il metodo di Cramer.
19. Il costo di spedizione di un pacco postale è composto da una quota fissa e dal peso del medesimo nel seguente modo: se il peso del pacco è inferiore ai 5 chili, il costo è di 10 euro, altrimenti se il peso del pacco è compreso tra 6 e i 10 kg allora il costo è pari a 15 euro, infine se il pacco supera i 10 chili allora il pacco costa 23 euro. Progettare un'ap-plicazione in grado di determinare il costo totale del pacco dati il peso del pacco e della quota fissa.
20. L'indice di massa corporea è un modo per dire se una persona in base ai suoi peso e altezza sia in forma o meno. In particolare dati peso e altezza, espressa in metri di una persona, l'indice viene calcolato nel seguente modo:
    $$IMC = \frac{Peso}{N^2}$$
    La diagnosi viene individuata dalle seguenti condizioni:
    Se IMC >=16 allora grave magrezza
    Se 16< IMC e IMC <18.50allora Sottopeso
    Se 18.50< =IMC e IMC< 25 allora Normopeso
    Se 25 <= IMC e IMC < 30allora Sovrappeso
    Se 30<= IMC e IMC<35 allora obeso di 1^ categoria
    Se 35<= IMC e IMC<40 allora obeso di 2^ categoria
    Se IMC > 40 allora obeso di 3^ categoria.
    Progettare un'applicazione in grado di individuare la diagnosi, dati in input peso e altezza di una persona.

21. Data la temperatura in un certo tipo di scala (Celsius, Fahreneit, Kelvin è possibile trasformarla nelle altre grazie alle seguenti formule:
Se la temperatura è in gradi Celsius allora le temperature in Fahreneit e in Kelvin sono date da:
TF = 32 + 1,8 * TC
TK = TC + 273,16
se la temperatura è in Fahreneit allora
TC = 5 * (TF - 32) /9
TK = TC + 237,16
se la temperatura è in kelvin allora
TC = TK - 273,16
TF = 32+9 * TC / 5
22. Il peso di un corpo dipende dall'ambiente in cui è collocato. Data la sua massa m, il peso è dato dalla formula: Peso = m* g dove g è l'accelerazione di gravità.
Sapendo che sulla Terra g=9.8 m/s^2; sulla Luna g=1.6 m/s^2; su Giove g = 26 m/s^2, progettare un'applicazione che forniti in input il nome dell'astro e la massa del corpo considerato, determini il relativo peso.
23. La scala Richter misura gli effetti di un terremoto sul territorio. La tabella di gravità del terremoto presente su wikipedia, al seguente indirizzo internet http://it.wikipedia.org/wiki/ScalaRichter descrive gli effetti di ciascuna scossa del sisma. Progettare un'applicazione in grado di acquisire l'intensità della scossa e di visualizzare il relativo effetto.

Iterazione

24. Un elenco di 10 numeri rappresenta gli incassi e le spese sostenute da un esercizio commerciale. Progettare un'applicazione in grado di determinare il totale degli incassi e quello delle spese, il bilancio e quindi stabilire se l'esercizio è in attivo o in perdita.
25. Le temperature dell'ultimo mese sono state raccolte in una lista. Progettare un'applicazione che individui il numero di volte in cui la temperatura è stata minore o uguale a zero.
26. Per una trasmissione televisiva sono previsti 5 ospiti, ciascuno con un proprio cachet. L'applicazione da progettare deve essere in grado di determinare la media tra tutti i cachet.
27. Dati i sette stipendi dei dipendenti di un piccola azienda, progetta un'applicazione in grado di determinare lo stipendio minore.
28. A scuola viene somministrato un quiz di 10 domande. Per ogni risposta corretta sono previsti 3 punti, se invece la risposta non lo è ne viene sottratto uno. I risultati per uno studente sono stati i seguenti:

1	2	3	4	5	6	7	8	9	10
G	G	S	S	G	G	S	G	G	S

Progettare un'applicazione in grado di determinare il punteggio raggiunto dallo studente.

29. La bolletta del gas arriva ogni due mesi a partire da gennaio. In un anno arrivano sei bollette:

Bolletta	Importo
1	45
2	30
3	40
4	55
5	45
6	60

Progettare un'applicazione in grado di individuare la bolletta con importo massimo.
Inoltre in base al numero della bolletta, individuare anche il nome dei due mesi associati alla stessa.

30. Per effettuare dei lavori di manutenzione, vengono chiamati cinque operai, i quali cominciano a e terminano di lavorare in orari differenti:

Operaio	Ora Inizio	Ora Fine
1	8.00	14.00
2	10.00	13.00
3	9.00	13.00
4	8.00	12.00
5	7.00	14.00

Progettare un'applicazione in grado di determinare:
il numero di ore lavorate e la paga di ciascun operaio; il costo totale del lavoro, sapendo che un'ora di lavoro costa 45 € e che per il materiale sono stati spesi 200 €.

31. Un rivista online è composta da 10 articoli ciascuno dei quali *pesa* un certo numero di KByte. Determinare l'articolo più "pesante" e il *peso* totale di "tutta la rivista".

11. Coding con "App Inventor"

Autori: **Grazia MARZIA** e **Angelo CAPUTO**

Competenze	Abilità	Conoscenze
Imparare a progettare e realizzare app con il linguaggio visuale App Inventor individuando le strategie appropriate per la soluzione dei problemi	*Sviluppare capacità di problemsolving Analizzare, progettare e realizzare applicazioni mobile con interfaccia grafica*	*Ambiente di sviluppo AppInventor*

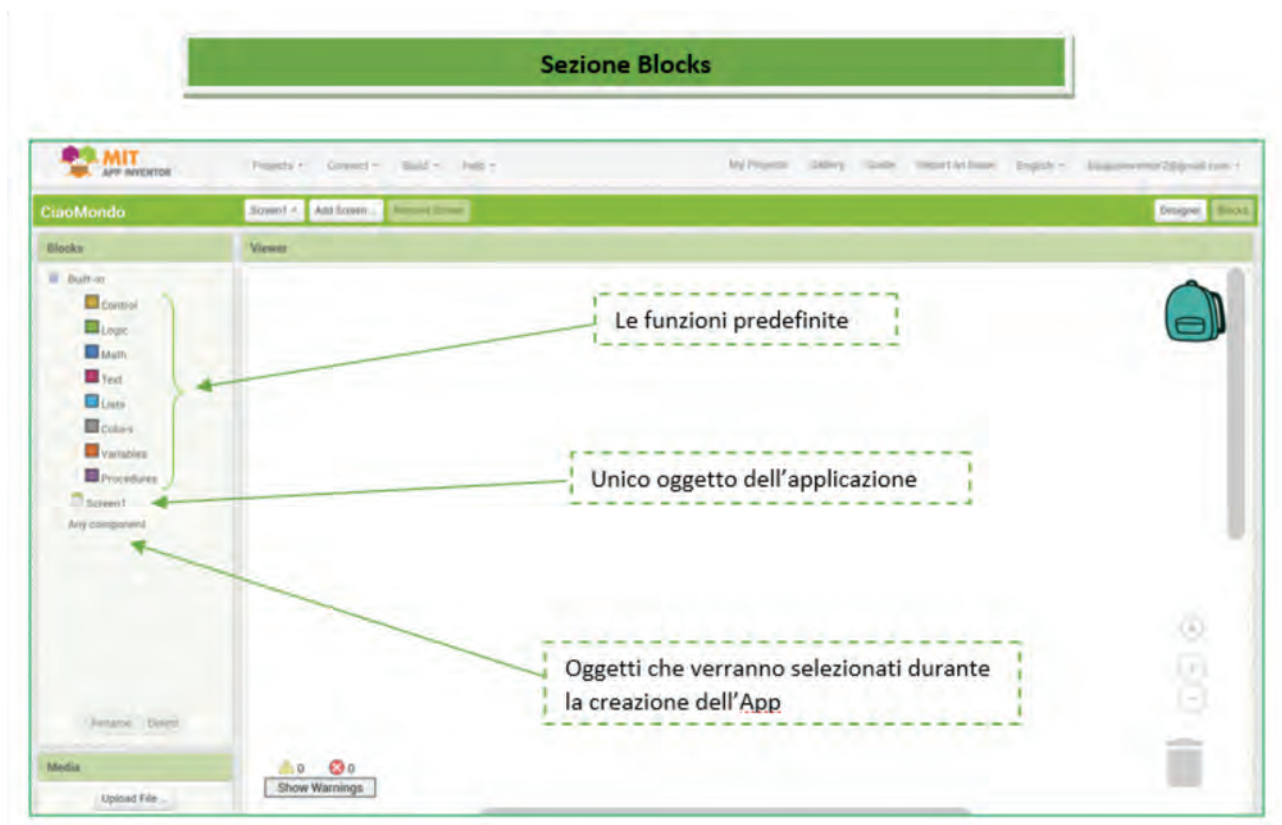
Creare con "APP Inventor"

App Inventor è una piattaforma web di programmazione che consente di creare in modo molto semplice applicazioni (app), per smartphone, con sistema operativo Android.
Questo ambiente di sviluppo remoto (cluod) permette, anche ai principianti, di creare e pubblicare facilmente app, attraverso tecniche di drag-and-drop e di programmazione visuale, in modo semplice ed intuitivo. La potenza dell'interfaccia grafica consente tuttavia anche ai programmatori esperti, di risparmiare tempo e velocizzare alcuni processi di sviluppo.
App Inventor è stato creato da Google nel 2009 come strumento di insegnamento per programmatori alle prime armi. Il gigante informatico ne annuncia la chiusura nell'anno 2011. Diviene quindi proprietà del Massachusetts Institute of Technology che porta avanti il progetto educativo denominandolo MIT App Inventor che, nel dicembre 2013, viene ridenominato App Inventor2.

L'ambiente di lavoro

La struttura generale dell'ambiente di sviluppo è costituita da due sezioni:

• una è riservata alla progettazione dell'aspetto grafico dell'app

- l'altra, è riservata allo sviluppo del codice vero e proprio dell'applicazione tramite l'assemblaggio dei blocchi funzionali che compongono il linguaggio di programmazione.

Lo sviluppo dell'App, dall'interfaccia grafica al codice a blocchi, non richiede alcun salvataggio in locale poiché il lavoro viene automaticamente salvato sul web server di App Inventor.

Per eseguire sullo smartphone o tablet Android il programma realizzato in App Inventor, è necessario aver preventivamente installato sul dispositivo una app, denominata "MIT AI2 Companion", reperibile gratuitamente sullo store di Google (Play Store). Dopo averla installata, è necessario assicurarsi che il PC e il dispositivo mobile condividano la medesima connessione WiFi per evitare la segnalazione di errore da parte dell'app.

Tuttavia è anche possibile eseguire l'applicazione dal PC attraverso un emulatore, oppure utilizzare un cavo USB per trasferire l'app da PC al dispositivo mobile.

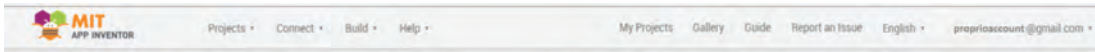
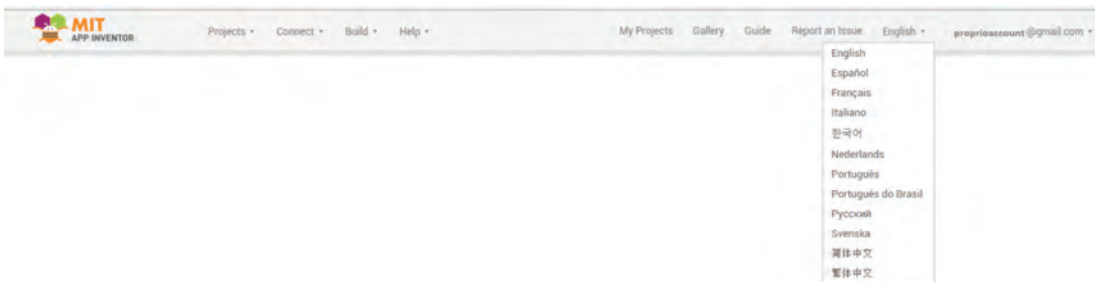
Effettuato il trasferimento, con uno dei metodi precedentemente indicati, sullo smartphone o tablet Android potremo eseguire l'app che compierà le operazioni programmate, al verificarsi degli eventi.

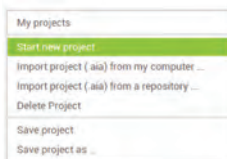
Cosa ci serve?

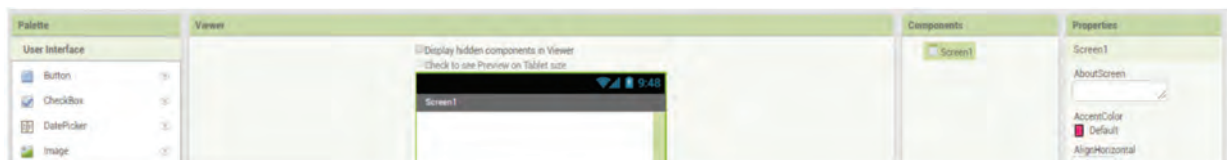
PC o notebook collegati a Internet
Un browser (preferibilmente Chrome)
Un account Google con un indirizzo e-mail del tipo ….@gmail.com
Uno smartphone o tablet

Esercitazione 1 l'app: "Parlami"

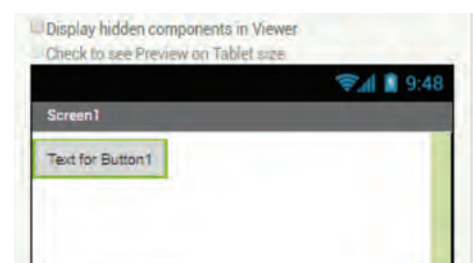
Vediamo come realizzare la nostra prima app che chiameremo progetto_uno.

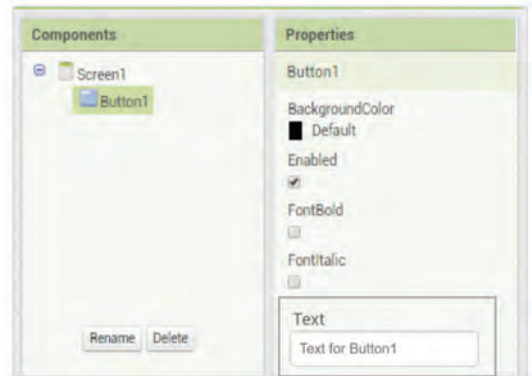
1. Entriamo con il nostro account Gmail
2. Andiamo all'indirizzo http://ai2.appinventor.mit.edu/
3. Rispondiamo ad alcune domande che chiede l'ambiente di sviluppo per applicazioni Android, creato da Google, ma ora di proprietà del Massachusetts Institute of Technology
4. Viene visualizzata l'intestazione nella versione inglese come mostrata in figura

5. È possibile cambiare lingua selezionandola dal menu a scelta multipla come in figura

6. Utilizzeremo l'impostazione inglese per rendere la disciplina trasversale con altre materie di indirizzo
7. Iniziamo il nostro primo progetto premendo sul menu a tendina Projects e selezionando "Start new project" come in figura

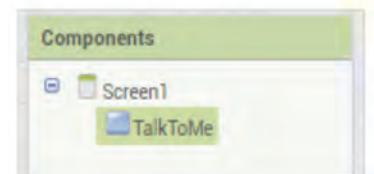
   My projects
   Start new project
   Import project (.aia) from my computer ...
   Import project (.aia) from a repository ...
   Delete Project
   Save project
   Save project as ...

8. Creiamo il nostro primo progetto dandogli un nome, nel nostro caso lo chiameremo "progetto_uno"

9. Ci si trova di fronte ad una videata suddivisa in quattro colonne così denominate: "Palette", "Viewer", "Components" e "Properties"

La sezione "Palette" contiene gli oggetti che saranno trascinati nella sezione "Viewer". Quest'ultima, rappresenta il display della nostra applicazione. Tutti gli oggetti presenti in "Viewer" saranno visibili nella sezione "Components". Le proprietà di ogni oggetto selezionato saranno visibili ed eventualmente modificabili nella sezione "Properties".

10. Creiamo la nostra prima App che abbiamo già denominato "progetto_uno" la cui funzionalità è quella di "parlarmi" **TalkToMe**

Trasciniamo dalla sezione "Palette" l'oggetto **Button** e posizioniamolo nella sezione **Viewer** nel riquadro **Screen1**. La videata dell'applicazione che visualizzeremo sul nostro **smartphone** è rappresentata proprio dallo Screen1, come in figura.

L'etichetta che l'applicazione mostra per default sull'oggetto **Button** è *Text for Button1*, tale oggetto è visibile nella sezione Components e alla propria destra, nella sezione "Properties" sono visibili le sue proprietà, vedi figura.

Cambiamo nella sezione Components, il nome dell'oggetto **Button**. Il nuovo nome, sarà quello con il quale ci riferiremo a questo oggetto all'interno del codice. Premiamo sul pulsante *Rename* della sezione Components. Comparirà una finestra dove sono visibili due box, *Old name* e *New name*, scriviamo nel box *New name* **TalkToMe** e premiamo il pulsante OK come in figura.

Osserviamo che nella sezione "Components" il nome dell'oggetto è stato modificato in TalkToMe, vedi figura.

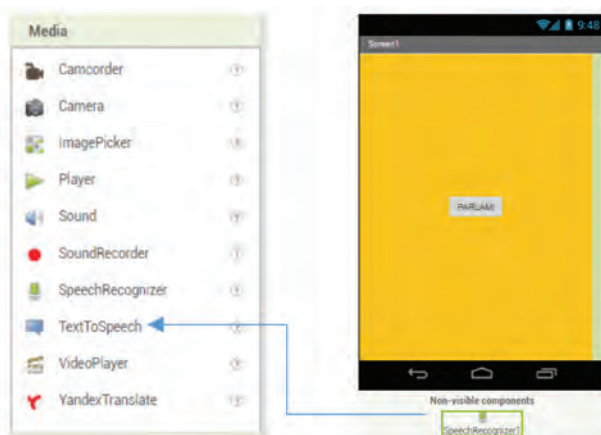
Adesso cambiamo il nome dell'etichetta che compare sul pulsante **Button**. Nella sezione "Properties", troviamo la proprietà *Text* dove osserviamo che compare la scritta *Text for Button1* e la modifichiamo con *PARLAMI*. Osserviamo che nella sezione "Viewer" il pulsante ha cambiato la propria etichetta. PARLAMI .

Finita la fase di progettazione dell'interfaccia della nostra applicazione, passiamo alla fase di programmazione. Tale passaggio si ottiene cliccando sul pulsante Blocks posto sopra la sezione "Properties". Designer Blocks .

Passando dall'ambiente Designer all'ambiente Blocks l'interfaccia cambia nel seguente modo

Nella precedente visualizzazione, esistono due sezioni, "Blocks" e "Viewer". Nella sezione Blocks ritroviamo gli oggetti che abbiamo già inserito nella fase di progettazione dell'interfaccia dell'applicazione.

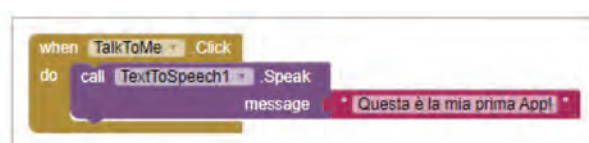
Per ritornare nella fase della progettazione grafica, basta cliccare sul pulsante Designer dove possiamo aggiungere ulteriori oggetti e modificare le relative proprietà.

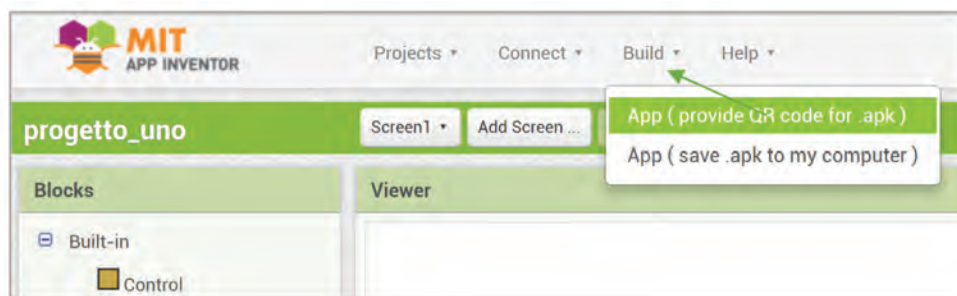
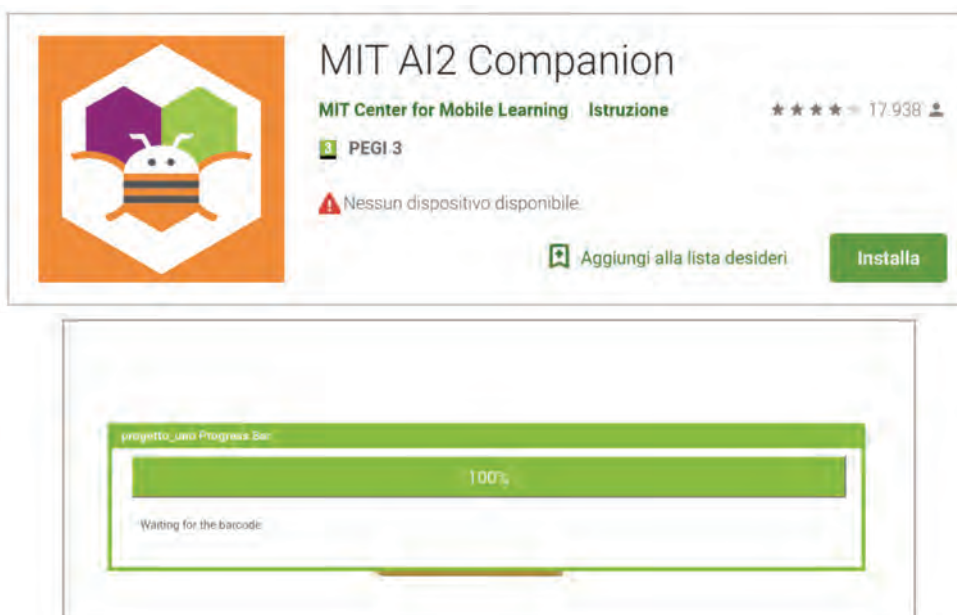
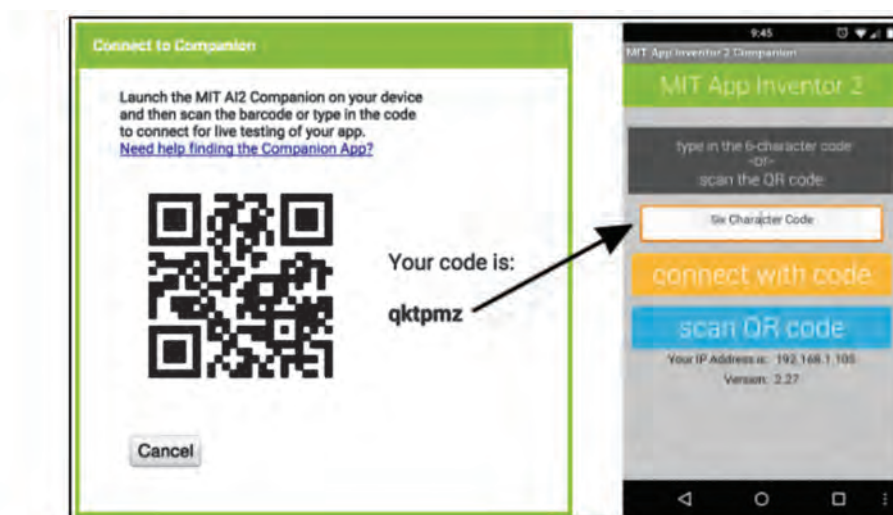
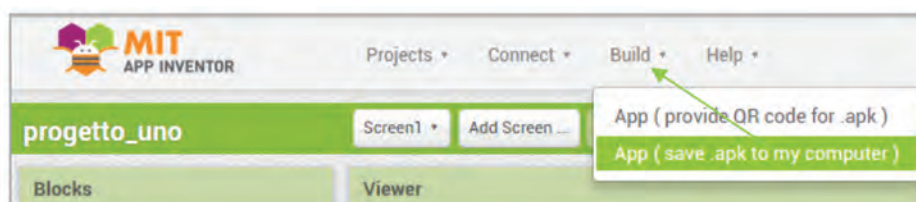
Quindi ritorniamo nella sezione "Designer" e modifichiamo le proprietà degli oggetti inseriti in precedenza ed aggiungiamo dei nuovi elementi. Cambiamo, nella sezione "Components", le proprietà AlignHorizontal, AlignVertical e BackgroundColor. Le variazioni apportate, modificheranno lo Screen1 come in figura.

Trasciniamo ora nel pannello **Viewer**, dalla sezione Palette, il componente **TextToSpeech**, individuandolo all'interno del gruppo **Media** come in figura. Tale componente trasformerà il testo presente in una casella di testo nel corrispondente audio.

Ora non resta che definire il comportamento della nostra app cioè specificarne i behavior. Nel nostro caso si tratta di riprodurre un messaggio vocale a seguito del clic sul pulsante PARLAMI.

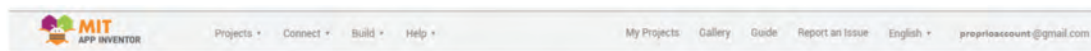
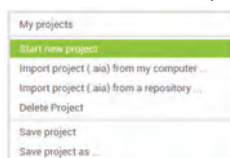
Passiamo quindi alla scheda **Blocks** della finestra principale. Selezioniamo la componente **TalkToMe** all'interno del pannello **Blocks**. Scegliamo e trasciniamo all'interno del pannello **Viewer**, il primo elemento tra quelli comparsi, cioè **When TalkToMe**.Click do come in figura.

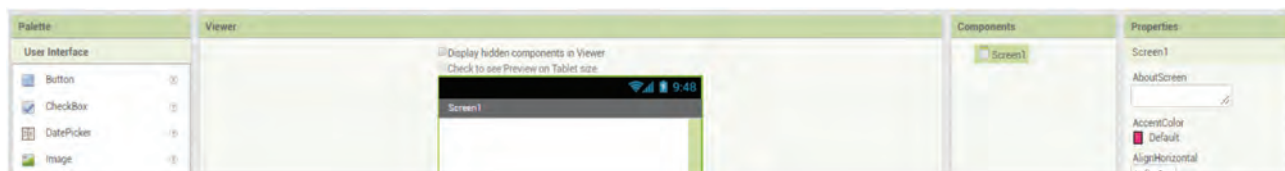
Selezioniamo il componente TextToSpeech1 e trasciniamo la proprietà call TextToSpeech1 Speak message all'interno del blocco **When TalkToMe**. Click do come in figura . Dobbiamo ora inserire il testo che verrà convertito dalla procedura TextToSpeech.

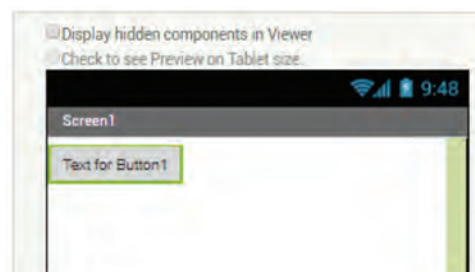
Sempre dall'area Blocks, clicchiamo il componente Text trasciniamo la seguente proprietà all'interno del blocco TextToSpeech. Otteniamo la seguente figura dopo aver digitato il messaggio che verrà convertito, nel nostro caso "Questa è la mia prima App!"

La fase conclusiva prevede il caricamento dell'applicazione appena creata sul nostro smartphone.

Creiamo il codice associato tramite "scan del QRCode"

L'App viene compilata per essere caricata sul nostro smartphone grazie a "MIT AI2 Companion" preventivamente installata sul cellulare.

Usando lo "scan QR code" oppure inserendo il codice creato si installa l'applicazione sul cellulare.

Inoltre se si vuole salvare la propria applicazione sul proprio computer, si deve selezionare da Build "App(save.apk to my computer), vedi figura.

Esercitazione 2 l'app: "Scrivi e ascolta"

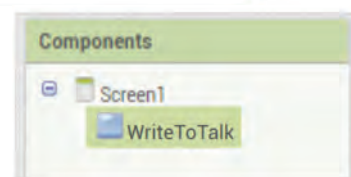
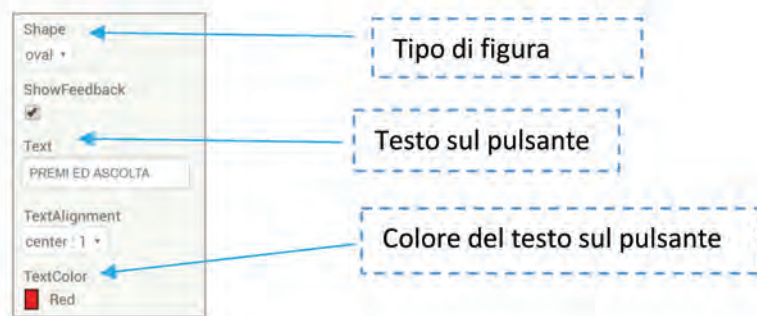
Vediamo come realizzare la nostra seconda app che chiameremo progetto_due.

2. Entriamo con il nostro account Gmail
3. Andiamo all'indirizzo http://ai2.appinventor.mit.edu/

4. Iniziamo il secondo progetto premendo sul menu a tendina Projects e selezioniamo "Start new project" come in figura

5. Creiamo il nuovo progetto assegnandogli il nome "progetto_due"

6. Ritroviamo la seguente videata

7. Creiamo la nostra seconda App che abbiamo già chiamato "progetto_due" la cui funzionalità è quella di **WriteToTalk** (scrivi nel box e ascolta)

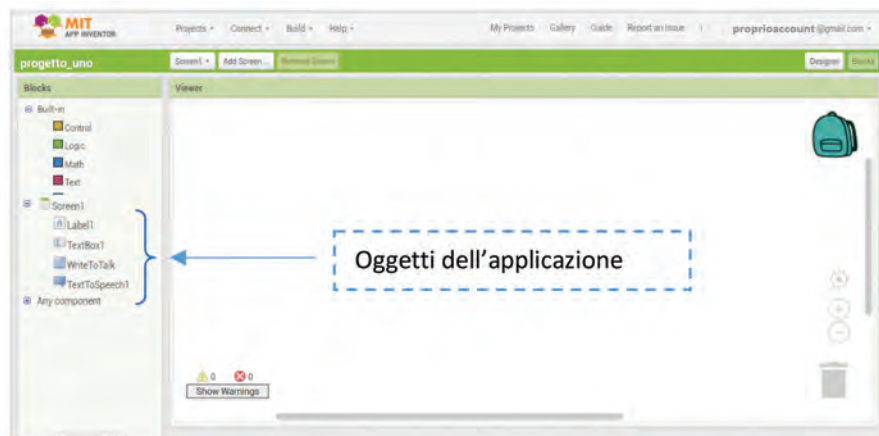
Trasciniamo dalla sezione "Palette" l'oggetto **Button** e posizioniamolo nella sezione **Viewer** nel riquadro **Screen1** che rappresenta la videata dell'applicazione che visualizzeremo sul nostro **smartphone** come in figura

Rinominiamo l'oggetto Button e scriviamo nel box *New name* **WriteToTalk** e premiamo il pulsante OK come in figura

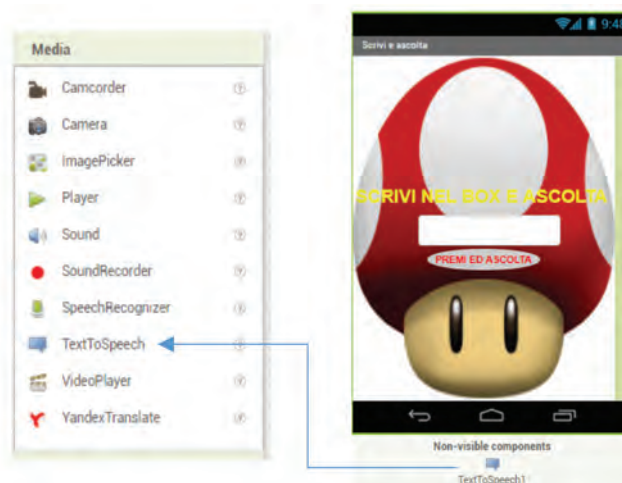
osserviamo nuovamente che, nella sezione "Components", il nome dell'oggetto è stato modificato in **WriteToTalk**, vedi figura.

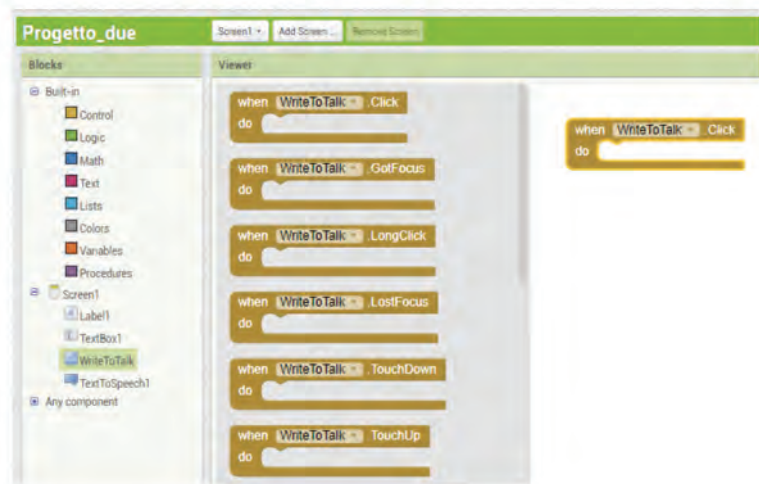
Adesso cambiamo il nome dell'etichetta che compare sul pulsante **Button.** Nella sezione "Properties", troviamo la proprietà *Text* dove notiamo la comparsa della scritta Text for Button1 e la modifichiamo con *PREMI ED ASCOLTA*. Osserviamo che nella sezione "Viewer" il pulsante ha cambiato la propria etichetta PREMI ED ASCOLTA

Per ottenere il risultato mostrato, sono state cambiate anche altre proprietà del pulsante “WriteToTalk” come in figura:

Per questo progetto serve un nuovo elemento della sezione Palette che si trova nel blocco User Interface: il componente è TexBo . Tale oggetto servirà per digitare il testo che sarà poi trasformato dalla funzione **TextToSpeech**, nel corrispondente sonoro.

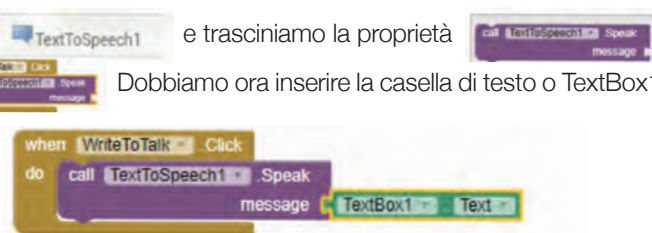
Inoltre utilizzeremo la proprietà Label per inserire un titolo alla nostra applicazione, “SCRIVI NEL BOX ED ASCOLTA”.

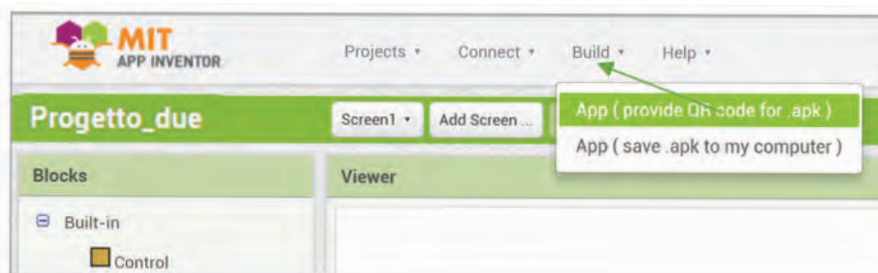
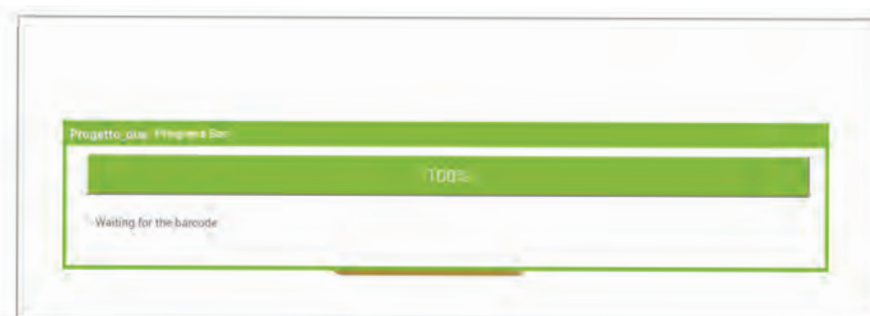
Finita la fase di progettazione dell’interfaccia della nostra applicazione, passiamo alla fase di programmazione, tale passaggio si ottiene cliccando sul pulsante Blocks posto sulla sezione “Properties”.

Passando dalla versione Designer alla versione Blocks l’interfaccia cambia nel seguente modo:

In questa videata, ritroviamo gli oggetti che utilizzeremo nella nostra dell’applicazione.

Ritorniamo nella fase della progettazione, cliccando su Designer e aggiungiamo la nuova funzione TextToSpeech. Trasciniamo ora nel pannello **Viewer**, dalla sezione Palette, il componente **TextToSpeech**, individuandolo all'interno del gruppo **Media** come in figura. Tale componente trasformerà il testo presente in una casella di testo, nel corrispondente audio

Ora non resta che definire il comportamento della nostra app cioè specificarne i **behavior**. Nel nostro caso riprodurre un messaggio vocale, inserito nella casella di testo TextBox1, a seguito del clic sul pulsante **PREMI ED ASCOLTA**.

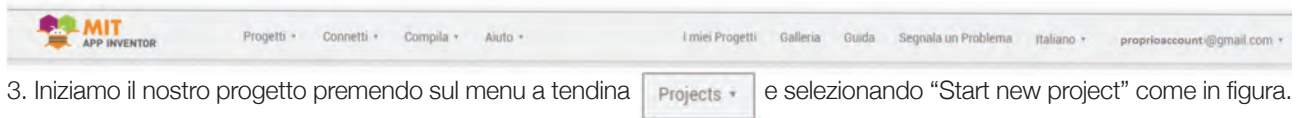
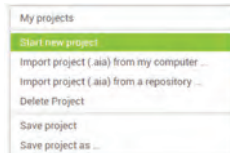
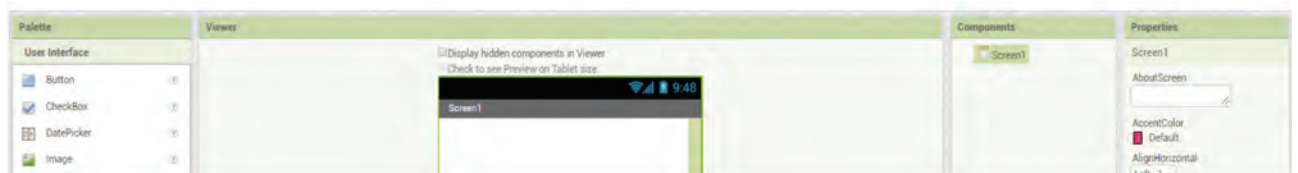
Passiamo nuovamente alla scheda **Blocks** della finestra principale. Selezioniamo la componente **WriteToTalk** all'interno del pannello **Blocks**. Selezioniamo e trasciniamo all'interno del pannello **Viewer**, il primo elemento tra quelli comparsi, vale a dire **When WriteToTalk**. Click do come in figura.

Selezioniamo il componente TextToSpeech1 e trasciniamo la proprietà all'interno del blocco **When WriteToTalk.Click do.** Dobbiamo ora inserire la casella di testo o TextBox1 nella funzione TextToSpeech.

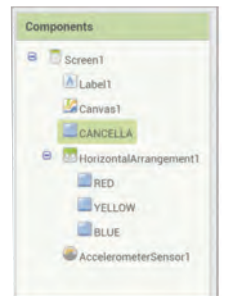
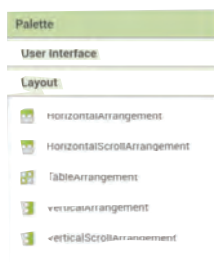
Otteniamo la seguente figura:

La fase conclusiva prevede il caricamento dell'applicazione appena creata sul nostro smartphone.

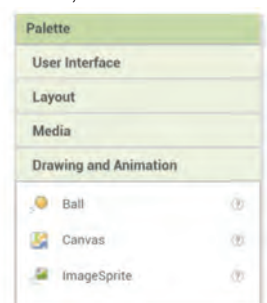
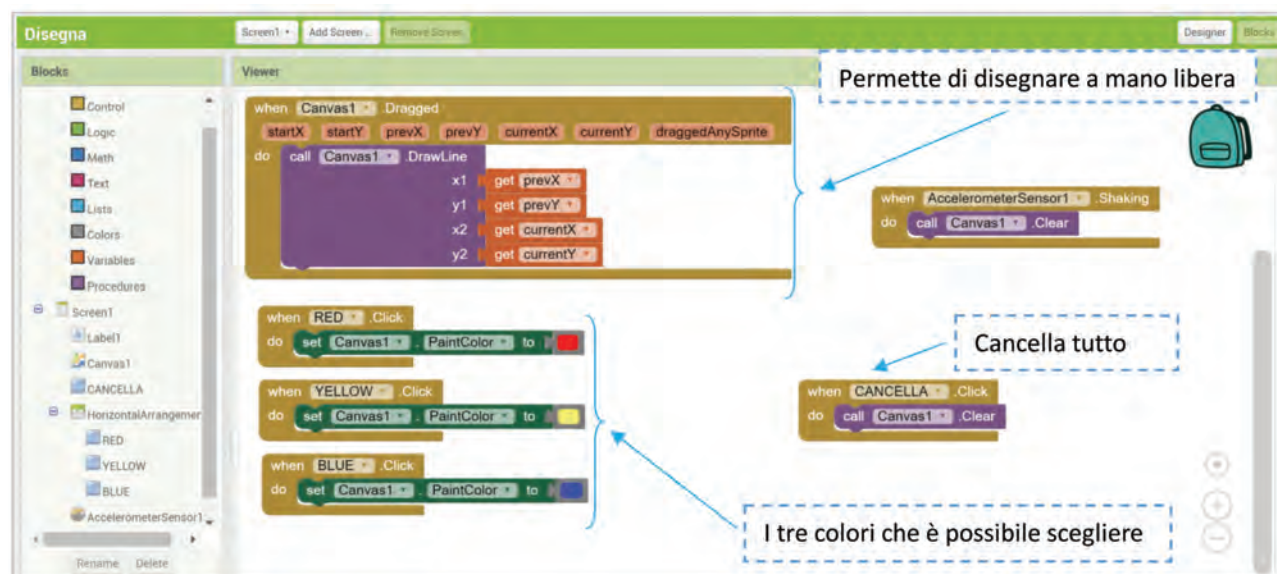
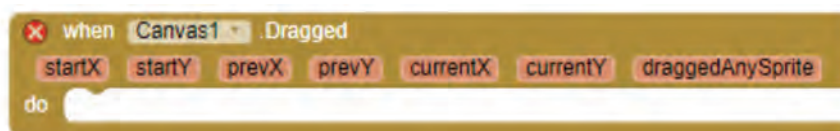
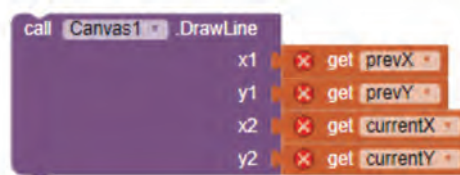
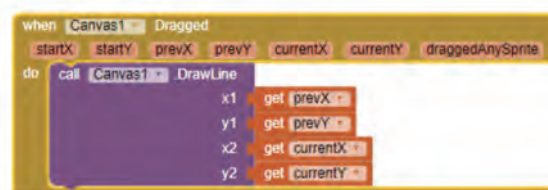
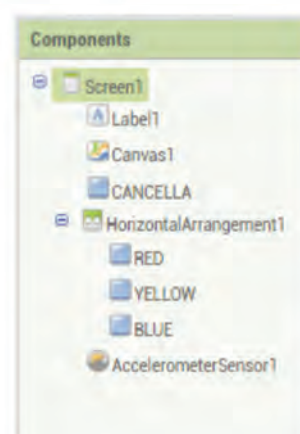
Creiamo il codice associato tramite "scan del QRCode"

L'App viene compilata per essere caricata sul nostro smartphone grazie a "MIT AI2 Companion"

Usando lo "scan QR code" oppure inserendo il codice creato si installa l'applicazione sul cellulare.

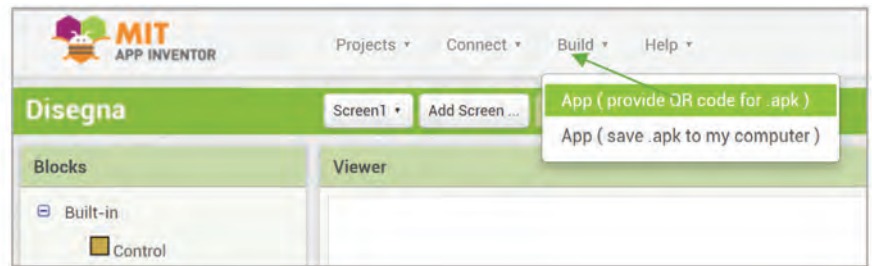
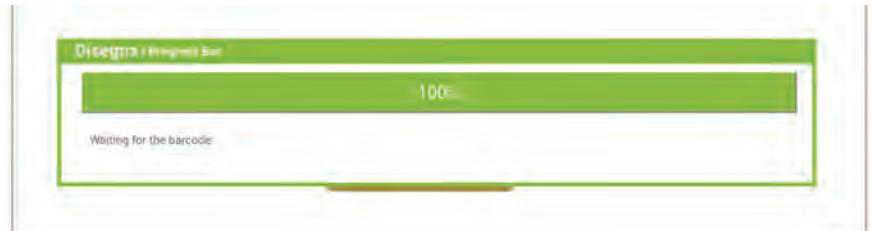
Esercitazione 3 l'app: "Scegli il colore e disegna"

Questa nuova App ci permetterà di scrivere e disegnare a mano libera sul nostro smartphone.
Sarà possibile scegliere il colore e ripulire la videata dello smartphone con un leggero movimento.

1. Entriamo con il proprio account GMail
2. Andiamo all'indirizzo http://ai2.appinventor.mit.edu/
3. Iniziamo il nostro progetto premendo sul menu a tendina Projects ▾ e selezionando "Start new project" come in figura.
4. Creiamo il nuovo progetto assegnandogli il seguente nome: Disegna.

5. Ritroviamo la videata ormai nota:

Trasciniamo dalla sezione "Palette" i seguenti Button, Label, Canvas, HorizontalArrangement e AccelerometerSensor. Posizioniamoli tutti nella sezione **Viewer** nel riquadro **Screen1** che rappresenta la videata dell'applicazione che visualizzeremo sul nostro **smartphone** e che ritroveremo nella sezione Components come si vede in figura.

I due nuovi oggetti che saranno presi in considerazione sono Non-visible components AccelerometerSensor1 componente non visibile la cui funzione è eseguire la cancellazione del testo al movimento del cellulare.
Mentre l'oggetto HorizontalScrollArrangement che si trova nella sezione Palette alla voce Layout, vedi figura

ci fornisce una struttura dove poter posizionare in maniera uniforme gli oggetti selezionati. Inoltre inseriamo i tre Button che daranno la possibilità di poter cambiare il colore del testo o dei grafici da visualizzare sul nostro smartphone. L'oggetto Canvas, presente nella sezione "Drawing and Animation", consente di definire una tela dove è possibile, muovendo il dito sullo schermo dellosmartphone, di disegnare o scrivere a mano libera Canvas vedi figura.

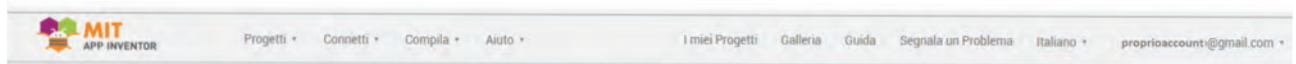
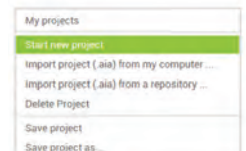
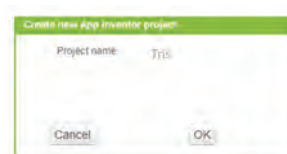
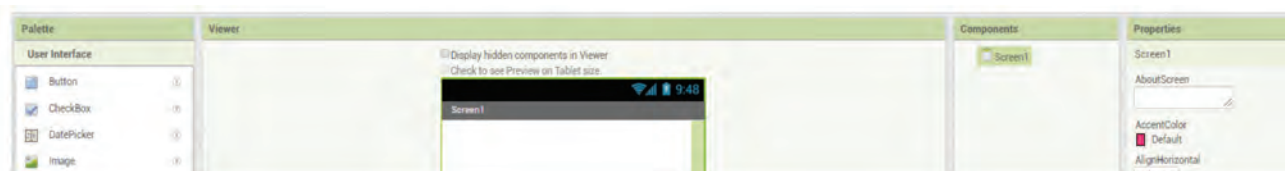
Finita la fase di progettazione dell'interfaccia della nostra applicazione, passiamo alla fase di programmazione, tale passaggio si ottiene cliccando sul pulsante Blocks posto sulla sezione "Properties" Designer Blocks .

Passando dalla versione Designer alla versione Blocks dobbiamo ora riprodurre il seguente codice:

Vediamo adesso come funziona la proprietà dell'oggetto Canvas Canvas , ovvero Canvas1.Dragged che ci darà la possibilità di disegnare sul nostro smartphone.

Dragged(number startX, number startY, number prevX, number prevY, number currentX, number currentY, boolean draggedSprite)
Quando l'utente sposta il dito sullo schermo del nostro smartphone, la posizione iniziale è rappresentata dai valori contenuti in (startX, startY), quella attuale è contenuta in (currentX, currentY), mentre in (prevX, prevY) sono contenute i valori precedenti all'evento di trascinamento. Si osserva che, nella posizione iniziale, i valori di (prevX, prevY) e (startX, startY) sono uguali.

Mentre con call Canvas1.DrawLine viene disegnata la linea corrispondente alle variazioni dei valori precedentemente memorizzati in (startX, startY, prevX, prevY, currentX, currentY).

Passando nuovamente alla visualizzazione Designer Designer Blocks vediamo il risultato della nostra interfaccia con i suoi relativi oggetti:

La fase conclusiva prevede il caricamento dell'applicazione appena creata sul nostro smartphone.

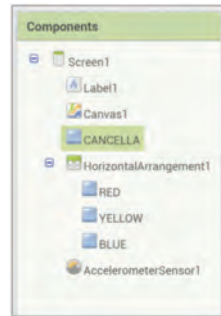
Creiamo il codice associato tramite “scan del QRCode”

L’App viene compilata per essere caricata sul nostro smartphone grazie a “MIT AI2 Companion”

Usando il QRCode del dell’App

ed installare sul proprio smartphone Android l’applicazione creata.

Esercitazione 4 l’app: “Il gioco del tris”

Questa App ci permetterà di giocare al gioco del tris, con la possibilità di scegliere un colore per tracciare i simboli X oppure O nella fase di gioco.

1. Entriamo con il nostro account GMail
2. Andiamo all’indirizzo http://ai2.appinventor.mit.edu/

3. Iniziamo il nostro nuovo progetto premendo sul menu a tendina e selezionando “Start new project” come in figura
4. Creiamo il nostro progetto assegnando il nome “Tris”
5. Ritroviamo la seguente videata

Gli oggetti utilizzati, sono quelli mostrati in figura:

I due oggetti, e HorizontalScrollArrangement sono già stati esaminati nel progetto precedente.

Finita la fase di progettazione dell'interfaccia della nostra applicazione, passiamo alla fase di programmazione, tale passaggio si ottiene cliccando sul pulsante Blocks posto sulla sezione "Properties".
Passando dalla versione Designer alla versione Blocks dobbiamo ora creare il seguente codice:

Passando nuovamente alla visualizzazione Designer vediamo il risultato della nostra interfaccia con i suoi relativi oggetti:

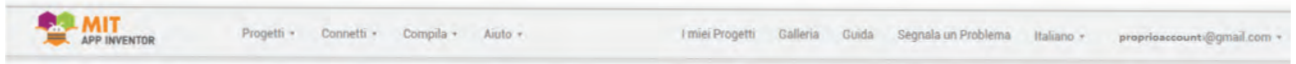
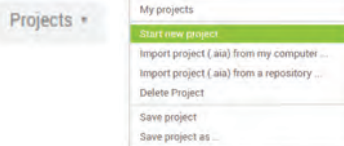
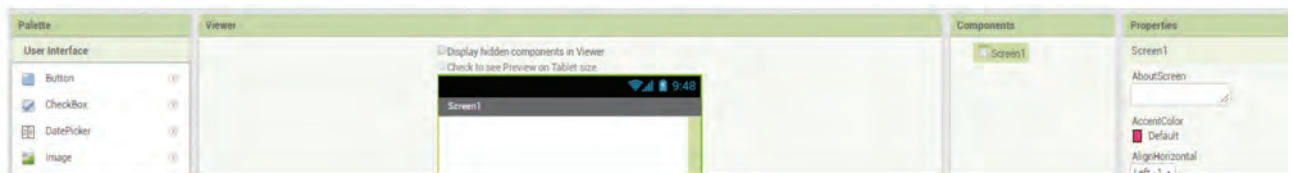
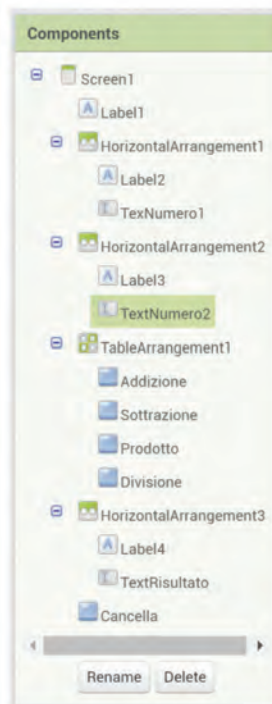
La fase conclusiva prevede il caricamento dell'applicazione appena creata sul nostro smartphone.
Creiamo il codice associato tramite "scan del QRCode"

L'App viene compilata per essere caricata sul nostro smartphone grazie a "MIT AI2 Companion" ed installare sul proprio smartphone Android l'applicazione creata.
SCARICA L'APP

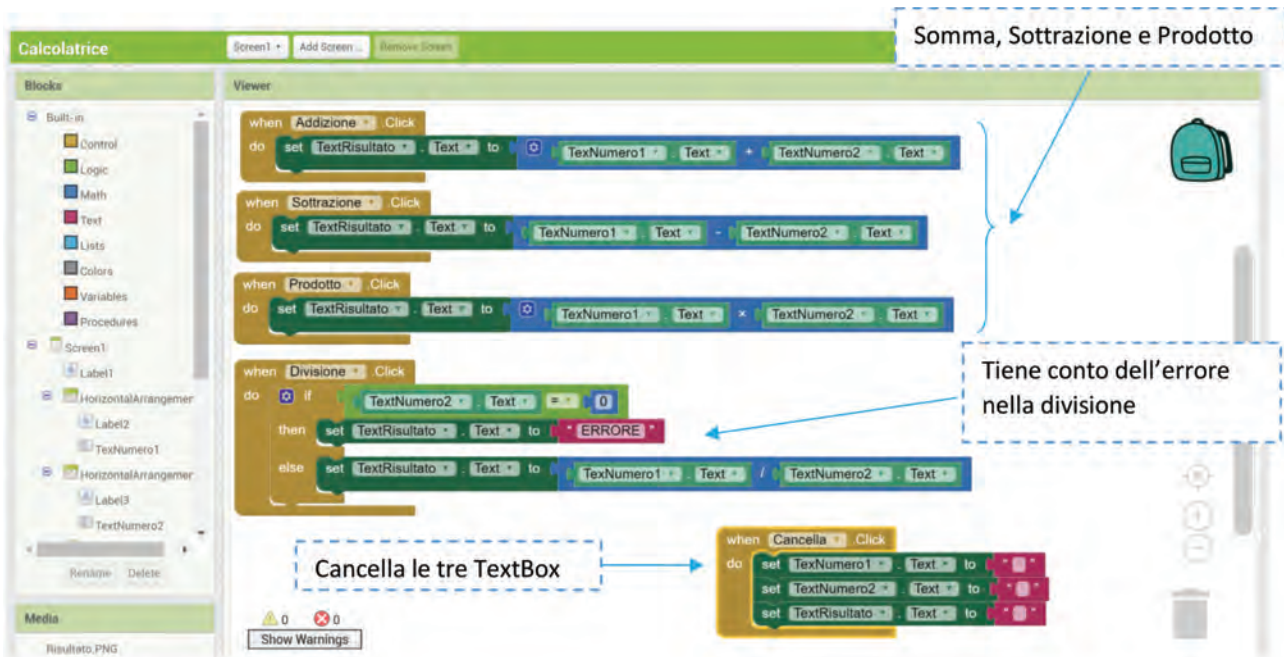
TRIS con App Inventor
http://qrbridge.me/app_tris

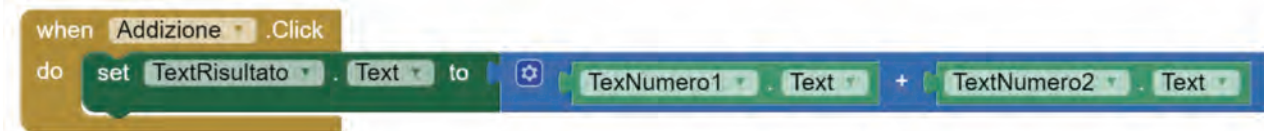
Esercitazione 5 l'app: "La calcolatrice"

Questa App è una calcolatrice che compie le quattro operazioni fondamentali evitando l'errore nella divisione per zero.

1. Entriamo con il nostro account Gmail
2. Andiamo all'indirizzo http://ai2.appinventor.mit.edu/

3. Iniziamo il nuovo progetto come in figura
4. Assegniamo il nome "Calcolatrice"
5. Ritroviamo la seguente videata

Gli oggetti utilizzati, sono quelli mostrati in figura:

In questa App sono previsti cinque Button: quattro daranno la possibilità di poter fare le quattro operazioni fondamentali (Addizione, Sottrazione, Prodotto e Divisione), il quinto Button ci consentirà di cancellare i dati presenti nei Textbox TextBox (TextNumero1, TextNumero2 e TextRisultato).

Finita la fase di progettazione dell'interfaccia della nostra applicazione, passiamo alla fase di programmazione, tale passaggio si ottiene cliccando sul pulsante Blocks posto sulla sezione "Properties". Designer Blocks

Passando dalla versione Designer alla versione Blocks dobbiamo ora riprodurre il seguente codice:

Prendiamo in esame le nuove strutture utilizzate per le quattro operazioni, in particolar modo sviluppiamo la funzione Addizione:

La logica è la seguente: quando si preme il pulsante Button Addizione, il risultato dei valori numerici contenuti in TexNumero1 e TextNumero2 vengono inseriti tramite la struttura Math somma nel TextRisultato ottenendo la seguente struttura:

Analogamente si procede per la sottrazione e moltiplicazione.

Relativamente all'operazione della divisione, ricordiamo che questa operazione non è possibile quando il denominatore è uguale a zero.

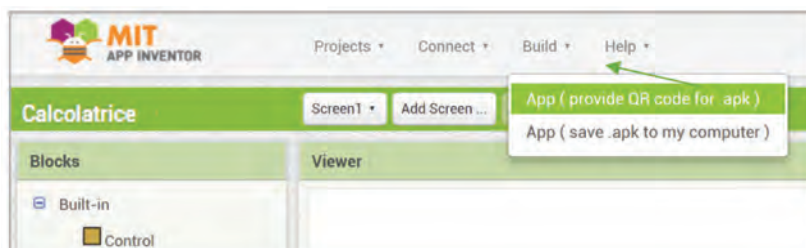
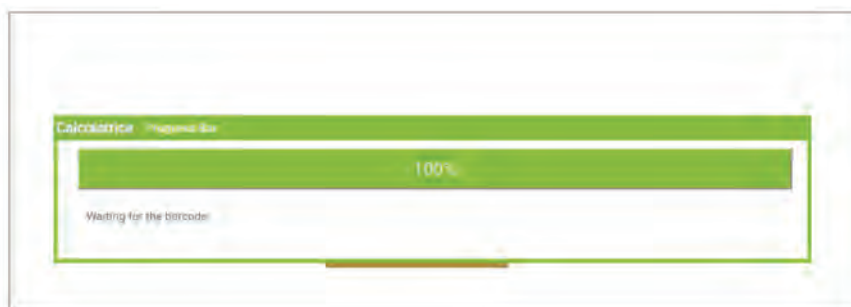
Per poter evitare questo errore dobbiamo utilizzare una struttura già vista con Excel, la funzione SE.

La sintassi di tale funzione, era cosi strutturata: =SE(Test; SE_Vero; SE_Falso). Con appinventor, la struttura equivalente è la seguente

Ottenendo la seguente struttura che dà un messaggio di errore quando al denominatore inseriamo il valore zero.

Passando nuovamente alla visualizzazione Designer

vediamo il risultato della nostra interfaccia con i suoi relativi oggetti:

La fase conclusiva prevede il caricamento dell'applicazione appena creata sul nostro smartphone.
Creiamo il codice associato tramite "scan del QRCode"

L'App viene compilata in un eseguibile

Usando il QRCode del dell'App

ed installare sul proprio smartphone Android l'applicazione creata.
Codice QR per scaricare l'App

Calcolatrice con App Inventor
http://qrbridge.me/bjny

Tabella dei Componenti User Interface

I componenti basilari: user interface

Button	Il classico bottone
CheckBox	Casella di scelta (true/false)
DatePicker	Selezione date
Image	Mostra un'immagine
Label	Una stringa testuale
ListPicker	Un bottone che apre una lista di scelte
ListView	Consente di creare una lista di elementi testuali
Notifier	Consente di creare notifiche
PasswordTextBox	Campo per l'inserimento di password (nascosta)
Screen	E' l'elemento genitore (lo schermo) che contiene tutti gli altri
Slider	Genera una barra con all'interno un cursore draggabile
Spinner	Apre un menu di scelta multipla
TextBox	Campo per l'inserimento di testo
TimePicker	Apre un pop-up per la selezione di un orario
WebViewer	Consente di aprire una URL remota

Componenti: media

Camcorder	Apre la videocamera integrata nel device per la registrazione di un video
Camera	Apre la videocamera integrata nel device per scattare una foto
ImagePicker	Consente di selezionare un'immagine tra quelle presenti nella galleria del dispositivo
Player	Consente di riprodurre un file audio e di controllare la vibrazione del device (consigliato per file audio di lunga durata)
Sound	Consente di riprodurre un file audio e di controllare la vibrazione del device (consigliato per file audio di breve durata)
SoundRecorder	Consente di accedere al microfono integrato nel device per effettuare una registrazione audio
SpeechRecognizer	Consente di attivare la funzionalità di riconoscimento vocale integrata in Android al fine di convertire un parlato in testo
TextToSpeech	Consente di trasformare un testo in un parlato attraverso un sintetizzatore vocale (tra i vari linguaggi è supportato anche l'italiano)
VideoPlayer	Consente di riprodurre un file video all'interno di un player dotato dei normali comandi attivabili al touch dell'utente
YandexTranslate	Consente di effettuare traduzioni in tempo reale attraverso le API offerte dal traduttore automatico di Yandex

Fonte: http://ai2.appinventor.mit.edu/reference/components/